Interpolatory surface subdivision

by

Håkon Neset

THESIS

for the degree

MASTER OF SCIENCE

(Master i Anvendt matematikk og mekanikk)

Faculty of Mathematics and Natural Sciences
University of Oslo

June 2011

Det matematisk- naturvitenskapelige fakultet
Universitetet i Oslo

Acknowledgments

I would like to express my gratitude to my supervisor Michael Floater for all the guidance and insightful discussions.

My appreciation goes to all of my teachers through many years of mathematical studies, and in special to Henrik Kirkegaard and Hallgeir Buset for giving me a deep interest in math and science from an early age.

I would also give thanks to my fellow students for all the laughs, and of course all the mathematical knowledge we have exchanged.

Last, but not the least, I would like to express my gratefulness to my family and Marte, for supporting and motivating me through all my work and studies.

Håkon Neset
Oslo, June 2011

Contents

Chapter 1

Introduction

When someone wants to build some kind of complicated structure, say, e.g. a house, a ship, an airplane or some monumental building, a vision of the final result is important. When humans first started to build structures, a simple sketch would do. But as we build more and more complex structures, a proper design process has become increasingly important. Today, we use computers to design the complex shapes of cars, airplanes and the like.

One of the more popular methods in CAGD (Computer Aided Geometric Design) is spline curves. This method has its offspring from shipbuilders centuries ago, when entire lofts above the shipyard where used to create one-to-one drawings of the ships. The shipbuilders made scaled sketches of the ships, then they plotted the key points on graph paper. These key points were then translated to one-to-one scale, and the lines between these points were then drawn using long, thin wooden strips, called "splines". These splines were passed through weights, called "ducks", placed at the key points, and created a minimum strain energy curve passing through the points.

Later, around the time of WWII, splines were also used in car and airplane design, and as time passed, mathematicians working in the air and automobile companies started developing a more mathematical description of splines. In the 1950s and 1960s much work was done by, among others, Paul de Casteljau at Citroën and Pierre Bézier at Renault.

The usual way of mathematically defining splines is not interpolatory, but if we go back to the origin of splines, the curves made using wooden splines and ducks interpolated the key points. So, with some work, one can make a mathematical description of interpolatory splines. But this introduces new problems, they require us to solve systems of equations, and in the case of surfaces, has some less desirable properties, as we shall se later.

Spline theory is also the basis of subdivision. It was discovered that one could refine the knot vector and control polygon of a spline, yet keep the

spline unchanged. This is done by dividing the knot intervals. The limit of this refinement makes the control polygon equal to the spline itself.

Later, in 1978, two new subdivision methods were published, today known as Catmull-Clark and Doo-Sabin subdivision. Both of these methods are for surfaces, and are based on respectively bi-cubic and bi-quadratic splines. They do not use the splines in the calculations of the limit surface, as both of these methods are iterated methods that only use the points in the initial mesh.

The problem with both of these subdivision methods, is that they are not interpolatory. This means that if we want to design something, e.g. by using a mesh of points as a representation of our design, and use one of these methods, the resulting surface would not pass through our original points. This makes designing with these methods difficult.

In the late 1980s interpolatory subdivision was discovered. This started with two independent articles published by Serge Dubuc in 1986 and Nira Dyn, David Levin and John Gregory in 1987. These articles described variants of what we today call the four point scheme.

The four point scheme is interpolatory by nature, and through the 1990s and the 2000s several variants solving various specific problems were published. In this thesis I will propose and study some new methods for interpolatory surface subdivision based on the four point scheme.

More about CAGD and its history can be found in [5], and for splines one can in addition look at [10].

Chapter 2

Interpolation for curves

2.1 Splines

In this section we will give a brief overview of splines, and how we use them to interpolate curves. But first we have to start with some definitions, and the first is the definition of B-splines.

Definition 1. *Let d be a nonnegative integer and let $\mathbf{t} = (t_j)$ be a knot vector, a nondecreasing sequence of real numbers, of length at least $d+2$. The jth B-spline of degree d with knots $\mathbf{t}$ is defined by*

$$B_{j,d,\mathbf{t}}(x) = \frac{x - t_j}{t_{j+d} - t_j} B_{j,d-1,\mathbf{t}}(x) + \frac{t_{j+1+d} - x}{t_{j+1+d} - t_{j+1}} B_{j+1,d-1,\mathbf{t}}(x) \qquad \forall x \in \mathbb{R}$$

where

$$B_{j,0,\mathbf{t}} = \begin{cases} 1, & \text{if } t_j \leq x < t_{j+1} \\ 0, & \text{otherwise} \end{cases}$$

Here, it is assumed that 0/0 is defined to be 0.

We might sometimes drop some of the subscripts, and write $B_{j,d}$ or B_j for $B_{j,d,\mathbf{t}}$. A knot has *multiplicity* m if it appears m times in the knot sequence, and knots with multiplicity one, two and three are called simple, double and triple knots.

Let us now define spline functions.

Definition 2. *Let $\mathbf{t} = (t_j)_{j=1}^{n+d+1}$ be a nondecreasing sequence of real numbers, i.e. a knot vector for a total of n B-splines. The linear space of all linear combinations of these B-splines is the spline space $\mathbb{S}_{d,\mathbf{t}}$ defined by*

$$\mathbb{S}_{d,\mathbf{t}} = \operatorname{span}\{B_{1,d}, \ldots, B_{n,d}\} = \Big\{ \sum_{j=1}^{n} c_j B_{j,d} \mid c_j \in \mathbb{R} \text{ for } 1 \leq j \leq n \Big\}$$

An element $g = \sum_{j=1}^{n} c_j B_{j,d}$ *of* $\mathbb{S}_{d,\mathbf{t}}$ *is called a spline function, or just a spline, of degree* d *with knots* $\mathbf{t}$*, and* $(c_j)_{j=1}^{n}$ *are called the B-spline coefficients of* g*.*

If we let d be one, two or three, we say that we have linear, quadratic or cubic splines (or B-splines).

A number of properties can be proved for B-splines and spline function, see e.g. chapter 2 and 3 in [10]. Here we will only state a few results. The first is about the local support of B-splines.

Lemma 1. *Let* d *be positive, and let* $\boldsymbol{t} = t_j$ *be a knot sequence. Then the* j*th B-spline on* $\boldsymbol{t}$ *depends only on the knots* t_j*,* $t_{j+1}, \ldots, t_{j+d+1}$ *and if* x *is outside the interval* $[t_j, t_{j+d+1})$ *then* $B_{j,d}(x) = 0$*.*

The next result is about the smoothness of splines.

Theorem 1. *A spline of degree* d *is* C^d *everywhere except at the knots, where it is* C^{d-m}*, where* m *is the multiplicity of the knot.*

This means that if we have a simple knot vector, and a cubic spline, the spline is C^2 everywhere, or if we have two equal knots, then a cubic spline is C^1 at this knot.

2.1.1 Cubic spline interpolation

Lets start with m points $a = x_1 < x_2 < \cdots < x_m = b$ with corresponding values $y_i = f(x_i)$. We want to find a spline that interpolate the given values, and belongs to $C^2[a, b]$. To do this we need two extra conditions to specify the interpolant uniquely, and we often use either the hermite boundary conditions $g'(a) = f'(a)$ and $g'(b) = f'(b)$ or the natural boundary conditions $g''(a) = g''(b) = 0$. We can now define a knot vector as

$$\boldsymbol{\tau} = (\tau_i)_{i=1}^{m+6} = (x_1, x_1, x_1, x_1, x_2, x_3, \ldots, x_{m-1}, x_m, x_m, x_m, x_m)$$

This gives us the spline space $\mathbb{S}_{3,\boldsymbol{\tau}}$. We now want to study the interpolation problem where we are given $(x_i, f(x_i))_{i=1}^{m}$ and, if needed, $f'(x_1)$ and $f'(x_m)$, where $x_i < x_{i+1}$ for all $i = 1, 2, \ldots, m-1$. We want to find a spline g in $\mathbb{S}_{3,\boldsymbol{\tau}}$ such that $g(x_i) = f(x_i)$ for $i = 1, 2, \ldots, m$ such that the boundary conditions hold.

It can be shown that this interpolation problem has a unique solution for both choices of boundary conditions and that this solution has, in some sense, the smallest second derivative and smallest curvature of all interpolants fulfilling the boundary conditions. As a matter of fact, this problem is the mathematical description of the wooden splines and ducks discussed in the introduction, and the smallest second derivative has a connection to the energy it takes to bend the wooden rulers. More about these properties of these interpolating splines can be found in section 5.4 of [10].

We will now do a brief study of the solution of this problem with the hermite boundary conditions, the natural boundary conditions are quite similar. What we want to find is a function

$$g = \sum_{i=1}^{m+2} c_i B_{i,3}$$

in $\mathbb{S}_{3,\boldsymbol{\tau}}$ such that

$$\begin{aligned} \sum_{j=1}^{m+2} c_j B_{j,3}(x_i) &= f(x_i) \quad \text{for } i = 1, 2, \ldots, m, \\ \sum_{j=1}^{m+2} c_j B'_{j,3}(x_i) &= f'(x_i) \quad \text{for } i = 1 \text{ and } m. \end{aligned} \tag{2.1}$$

This is a linear system of $m+2$ equations in the $m+2$ unknown B-spline coefficients. From the local support of the B-splines, only a few unknowns appear in each equation. By letting the boundary conditions be the first and last equation, we get

$$\boldsymbol{Ac} = \begin{pmatrix} \alpha_1 & \gamma_1 & & & \\ \beta_2 & \alpha_2 & \gamma_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_{m+1} & \alpha_{m+1} & \gamma_{m+1} \\ & & & \beta_{m+2} & \alpha_{m+2} \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{m+1} \\ c_{m+2} \end{pmatrix} = \begin{pmatrix} f'(x_1) \\ f(x_1) \\ \vdots \\ f(x_m) \\ f'(x_m) \end{pmatrix} = \boldsymbol{f}$$

where the elements of $\boldsymbol{A}$ is given by

$$\begin{aligned} \alpha_1 &= B'_{1,3}(x_1), & \alpha_{m+2} &= B'_{m+2,3}(x_m), \\ \gamma_1 &= B'_{2,3}(x_1), & \beta_{m+2} &= B'_{m+1,3}(x_m), \\ \beta_{i+1} &= B_{i,3}(x_i), & \alpha_{i+1} &= B_{i+1,3}(x_i), & \gamma_{i+1} &= B_{i+2,3}(x_i) \end{aligned}$$

As we can see, the cubic spline interpolation is not a local method, as all of the data points are needed to calculate the spline interpolant. In the next sections we will discuss interpolatory subdivision, which is almost as smooth as this method, but is a local method.

2.2 Interpolatory subdivision

In interpolatory subdivision we are given a sequence of values f_k at an increasing sequence of grid points x_k, where $k \in \mathbb{Z}$. These values can be the values of a function f at x_k, or the pair (x_k, f_k) can be seen as a point added to a sketch by a designer. Either way, if we draw out straight lines between the points (x_k, f_k) we get a polygon, which we denote by g_0.

A polygon is continuous, but does not have a continuous derivative, so it does not look "smooth". The main purpose in subdivision is going from this polygon, g_0, to a smooth curve passing through the same initial points (x_k, f_k). We do this by an iterative method, for each step we find a new polygon which is in some sense closer to a smooth curve than the previous polygon. In each step we do this by finding new grid points between the old ones, and then calculate new values at the new grid points using the old values.

In a more formal definition, we set $x_{0,k} = x_k$ and $g_{0,k} = f_k$ for all k. Then, for each $j \geq 0$, we set $x_{j+1,2k} = x_{j,k}$ and choose $x_{j+1,2k+1}$ such that

$$x_{j,k} < x_{j+1,2k+1} < x_{j,k+1}$$

Similarly, we set $g_{j+1,2k} = g_{j,k}$, and then use a rule to calculate $g_{j+1,2k+1}$ using some of the values $g_{j,r}$ and $x_{j+1,s}$, $r, s \in \mathbb{Z}$.

A simple such rule could be

$$g_{j+1,2k+1} = g_{j,k} + (g_{j,k+1} - g_{j,k}) \frac{x_{j+1,2k+1} - x_{j,k}}{x_{j,k+1} - x_{j,k}}$$

This makes the new point $(x_{j+1,2k+1}, g_{j+1,2k+1})$ lie on the straight line between the points $(x_{j,k}, g_{j,k})$ and $(x_{j,k+1}, g_{j,k+1})$, which is of little use. More interesting rules will be discussed in the next sections.

After each step we get a set of points $(x_{j,k}, g_{j,k})$. We denote a linear interpolant to these points by g_j. If we study the sequence of polygons g_j, $j = 0, 1, 2, \ldots$ and the limit of this sequence, it can be shown that the limit is C^1 for good choices for the rules for new grid points and new values. The four point scheme is the perhaps the simples such choice.

2.3 Four point scheme

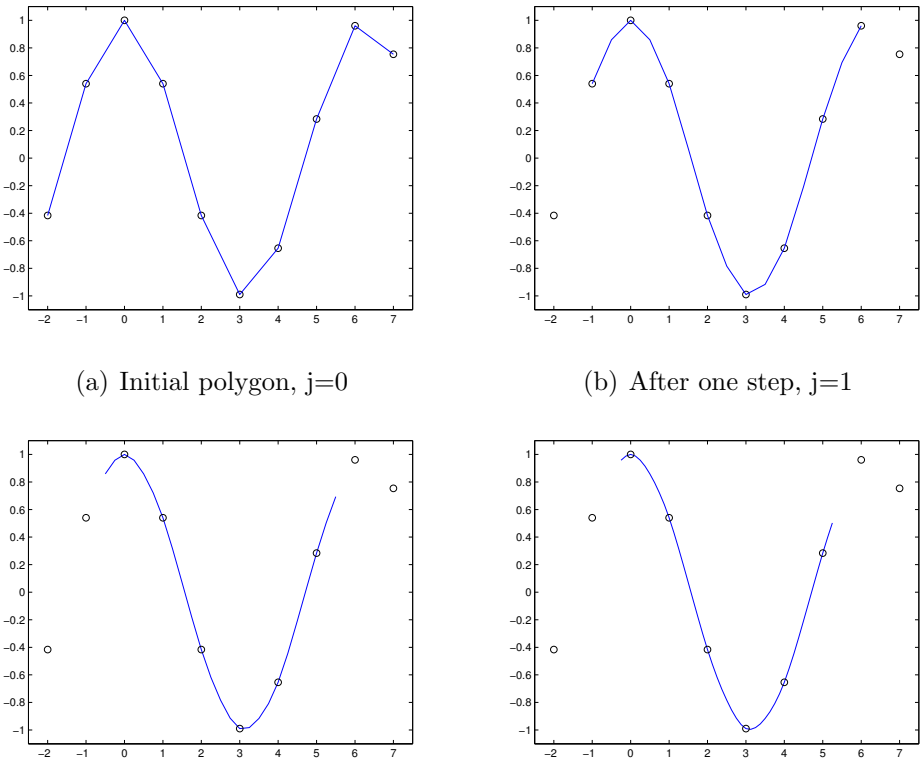
In 1986 Serge Dubuc [2] published one of the first articles about interpolatory subdivison, and a year later Dyn et al. [4] published an independent study. Both are variants of the four point scheme.

If we are given a sequence of values f_k where $k \in \mathbb{Z}$, we want to find a function g such that g is an interpolant to these values at the integers, so $g(k) = f_k$. We start by setting $g_{0,k} = f_k$ and $x_{0,k} = k$ for all $k \in \mathbb{Z}$. Then the first polygon is the linear interpolant g_0 to the data at k. Now, we calculate new, finer polygons by the four point rule. For $j = 0, 1, 2, \ldots$ we choose the new grid points as the midpoints between the old ones, so $x_{j+1,2k+1} = (x_{j,k} + x_{j,k+1})/2$, and the new values are defined by

$$\begin{aligned} g_{j+1,2k} &= g_{j,k} \\ g_{j+1,2k+1} &= -\frac{1}{16} g_{j,k-1} + \frac{9}{16} g_{j,k} + \frac{9}{16} g_{j,k+1} - \frac{1}{16} g_{j,k+2} \end{aligned} \tag{2.2}$$

Now, we let g_j be the piecewise linear interpolant to $(x_{j,k}, g_{j,k})$. The grid points are $x_{j,k} = 2^{-j}k$, wich are the integers for $j = 0$, the half integers for $j = 1$, and so on. These grid points are called the dyadic points, and the grid is called a dyadic grid.

The values $-\frac{1}{16}, \frac{9}{16}, \frac{9}{16}$ and $-\frac{1}{16}$ are called the mask of the subdivision scheme. These values come from cubic interpolation on the points -1, 0, 1 and 2, they are the values at $\frac{1}{2}$ of the four cubic Lagrange polynomials that have the value 1 at one of the points -1,0,1 and 2, and zero at the others. Hence, if $f_k = p(k)$ for some cubic polynomial p, then $g_{j,k} = p(2^{-j}k)$ for all $j = 0, 1, \ldots$ and $k \in \mathbb{Z}$. This means that the scheme has *cubic precision*.



(a) Initial polygon, j=0

(b) After one step, j=1

(c) After two steps, j=2

(d) After three steps, j=3

Figure 2.1: Three steps of the four point scheme

2.3.1 Convergence

We now want to study the sequence of polygons $g_0, g_1, \ldots$. We set

$$g = \lim_{j\to\infty} g_j$$

To show that g exists and is continuous, we need to show that the sequence of g_j is a Cauchy sequence in the max norm,

$$||f||_\infty = \max_x |f(x)|$$

Hence, we want to show that for any $\epsilon > 0$ there is some N such that for $i, j > N$,

$$||g_i - g_j||_\infty < \epsilon \tag{2.3}$$

To do this we use the following lemma.

Lemma 2. *If there are positive constants C and $\lambda < 1$ such that*

$$||g_{j+1} - g_j||_\infty \leq C\lambda^j, \qquad j = 0, 1, 2, \ldots \tag{2.4}$$

then g_j is a Cauchy sequence.

Proof. Assume $i > j > N$, and that (2.4) holds. Then

$$\begin{aligned}
||g_i - g_j||_\infty &\leq ||g_i - g_{i-1}||_\infty + ||g_{i-1} - g_{i-2}||_\infty + \ldots + ||g_{j+1} - g_j||_\infty \\
&\leq C\lambda^{i-1} + C\lambda^{i-2} + \ldots + C\lambda^j \\
&= C\lambda^j(\lambda^{i-1-j} + \lambda^{i-2-j} + \ldots + \lambda + 1) \\
&\leq C\lambda^j(\ldots + \lambda^2 + \lambda + 1) \\
&= C\lambda^j/(1-\lambda) \leq C\lambda^N/(1-\lambda)
\end{aligned}$$

Hence (2.3) holds if we take N large enough such that $C\lambda^N/(1-\lambda) \leq \epsilon$. □

We are now ready to prove the existence and continuity of g.

Theorem 2. *The sequence $g_0, g_1, g_2, \ldots$ has a continuos limit*

$$g(x) = \lim_{j\to\infty} g_j(x) \qquad\qquad x \in \mathbb{R}$$

Proof. The difference between the polygons g_j and g_{j+1} is itself a polygon at level $j + 1$. Hence it must attain its absolute maximum value at one of the points $x_{j+1,k} = 2^{-(j+1)}k$. We let $d_{j+1} = g_{j+1} - g_j$ be the difference polygon. We now observe

$$||g_{j+1} - g_j||_\infty = ||d_{j+1}||_\infty = \max_k |d_{j+1,k}|$$

where $d_{j+1,k} := d_{j+1}(x_{j+1,k})$. If we look at the values at $x_{j+1,2k}$, the difference is

$$\begin{aligned} |d_{j+1,2k}| &= |g_{j+1}(x_{j+1,2k}) - g_j(x_{j+1,2k})| \\ &= |g_{j+1,2k} - g_j(x_{j,k})| = |g_{j+1,2k} - g_{j,k}| = 0 \end{aligned}$$

by the definition in (2.2). The values at the points $x_{j+1,2k+1}$ are

$$\begin{aligned} |d_{j+1,2k+1}| &= |g_{j+1}(x_{j+1,2k+1}) - g_j(x_{j+1,2k+1})| \\ &= \left| g_{j+1,2k+1} - \frac{g_{j,k} + g_{j,k+1}}{2} \right| \end{aligned}$$

From (2.2) we get

$$\begin{aligned} |d_{j+1,2k+1}| &= \left| g_{j+1,2k+1} - \frac{1}{2}(g_{j,k} + g_{j,k+1}) \right| \\ &= \left| -\frac{1}{16} g_{j,k-1} + \frac{9}{16} g_{j,k} + \frac{9}{16} g_{j,k+1} - \frac{1}{16} g_{j,k+2} - \frac{1}{2}\left(g_{j,k} + g_{j,k+1}\right) \right| \\ &= \left| \frac{1}{16} \Big(- g_{j,k-1} + g_{j,k} + g_{j,k+1} - g_{j,k+2} \Big) \right| \end{aligned}$$

This can be written as the difference between two successive values $\delta_{j,k} := g_{j,k} - g_{j,k-1}$:

$$\begin{aligned} |d_{j+1,2k+1}| &= \frac{1}{16} \left| \Big(g_{j,k} - g_{j,k-1} - (g_{j,k+2} - g_{j,k+1}) \Big) \right| \\ &= \frac{1}{16} \left| \delta_{j,k} - \delta_{j,k+2} \right| \end{aligned}$$

Now, since $d_{j+1,2k} = 0$, we get

$$||g_{j+1} - g_j||_\infty = ||d_{j+1}||_\infty = \max_k |d_{j+1,k}| \le \frac{1}{8} \max_k |\delta_{j,k}| \tag{2.5}$$

If we can find C' and λ such that

$$\max_k |\delta_{j,k}| \le C' \lambda^j, \qquad j = 0, 1, 2, \ldots$$

then we can apply Lemma 1 with $C = C'/8$. From (2.2) we get

$$\begin{aligned} \delta_{j+1,2k} &= g_{j+1,2k} - g_{j+1,2k-1} \\ &= g_{j,k} + \frac{1}{16} g_{j,k-2} - \frac{9}{16} g_{j,k-1} - \frac{9}{16} g_{j,k} + \frac{1}{16} g_{j,k+1} \\ &= -\frac{1}{16} \delta_{j,k-1} + \frac{1}{2} \delta_{j,k} + \frac{1}{16} \delta_{j,k+1} \qquad (2.6) \\ \delta_{j+1,2k+1} &= g_{j+1,2k+1} - g_{j+1,2k} \\ &= -\frac{1}{16} g_{j,k-1} + \frac{9}{16} g_{j,k} + \frac{9}{16} g_{j,k+1} - \frac{1}{16} g_{j,k+2} - g_{j,k} \\ &= \frac{1}{16} \delta_{j,k} + \frac{1}{2} \delta_{j,k+1} - \frac{1}{16} \delta_{j,k+2} \qquad (2.7) \end{aligned}$$

We observe that

$$\max_k |\delta_{j+1,k}| \leq \frac{5}{8} \max_k |\delta_{j,k}| \tag{2.8}$$

We can now apply Lemma 1 with $C = \max_k |\delta_{0,k}|/8$ and $\lambda = 5/8$, and the theorem holds. □

2.3.2 Smoothness

We now want to consider the smoothness of the limit function g. Remember the values $x_{j,k} = 2^{-j}k$ from the definition of g_j, they are the grid points of g_j, the piecewise linear interpolant to $(x_{j,k}, g_{j,k})$.

To work on the smoothness of g we define the divided differences:

$$g_{j,k}^{[0]} := g_{j,k}$$

and, for $m \geq 1$,

$$g_{j,k}^{[m]} := \frac{g_{j,k+1}^{[m-1]} - g_{j,k}^{[m-1]}}{2^{-j}(k+m) - 2^{-j}k} = \frac{\Delta g_{j,k}^{[m-1]}}{2^{-j}m} = \frac{2^j}{m} \Delta g_{j,k}^{[m-1]}$$

where $\Delta g_{j,k}^{[m]} := g_{j,k+1}^{[m]} - g_{j,k}^{[m]}$.

Theorem 3. *The limit function of Theorem 1 is C^1.*

This proof is based on the article by Dyn et al. [4], and the lecture notes of Floater [8].

Proof. We let $g_j^{[1]}$ be the piecewise linear interpolant to the data $(x_{j,k}, g_{j,k}^{[1]})$. To prove the result we want to show (i) that the sequence of polygons $g_j^{[1]}$ has a continuous limit and (ii) that this limit is the derivative of g. To do this we show

(i) the sequence of polygons $g_j^{[1]}$ has a continuous limit

$$g^{[1]}(x) := \lim_{j\to\infty} g_j^{[1]}(x), \qquad x \in \mathbb{R}$$

(ii) that

$$g(x) - g(0) = \int_0^x g^{[1]}(y)\,\mathrm{d}y, \qquad x \in \mathbb{R} \tag{2.9}$$

which implies that g is differentiable with $g'(x) = g^{[1]}(x)$.

We start by (i), and we use the same approach as in the proof of Theorem 2, we want to show that $g_j^{[1]}$ is a Cauchy sequence in the max norm, and we use Lemma 1 to prove this.

For $\Delta g_{j,k} = g_{j,k+1} - g_{j,k}$ we can define a subdivision scheme exactly as we did for $\delta_{j+1,k}$ in (2.6) and (2.7):

$$\begin{aligned}\Delta g_{j+1,2k} &= \frac{1}{16}\Delta g_{j,k-1} + \frac{1}{2}\Delta g_{j,k} - \frac{1}{16}\Delta g_{j,k+1}\\ \Delta g_{j+1,2k+1} &= -\frac{1}{16}\Delta g_{j,k-1} + \frac{1}{2}\Delta g_{j,k} + \frac{1}{16}\Delta g_{j,k+1}\end{aligned}$$

Since $g_{j,k}^{[1]} = 2^j \Delta g_{j,k}$, we find

$$\begin{aligned}g_{j+1,2k}^{[1]} &= \frac{1}{8}g_{j,k-1}^{[1]} + g_{j,k}^{[1]} - \frac{1}{8}g_{j,k+1}^{[1]}\\ g_{j+1,2k+1}^{[1]} &= -\frac{1}{8}g_{j,k-1}^{[1]} + g_{j,k}^{[1]} + \frac{1}{8}g_{j,k+1}^{[1]}\end{aligned} \tag{2.10}$$

We now denote $d_{j+1}^{[1]} = g_{j+1}^{[1]} - g_j^{[1]}$, and let $d_{j,k}^{[1]} = d_j^{[1]}(2^{-j}k)$. Similar to the proof of Theorem 1, the maximum of $d_{j+1}^{[1]}$ is at one of the points $x_{j+1,k}$. This gives

$$\begin{aligned}d_{j+1,2k}^{[1]} &= g_{j+1,2k}^{[1]} - g_{j,k}^{[1]}\\ d_{j+1,2k+1}^{[1]} &= g_{j+1,2k+1}^{[1]} - \frac{1}{2}\Big(g_{j,k}^{[1]} + g_{j,k+1}^{[1]}\Big)\end{aligned}$$

By (2.10) we get

$$\begin{aligned}d_{j+1,2k}^{[1]} &= \frac{1}{8}g_{j,k-1}^{[1]} - \frac{1}{8}g_{j,k+1}^{[1]}\\ d_{j+1,2k+1}^{[1]} &= -\frac{1}{8}g_{j,k-1}^{[1]} + \frac{1}{2}g_{j,k}^{[1]} - \frac{3}{8}g_{j,k+1}^{[1]}\end{aligned}$$

We can rewrite this as

$$\begin{aligned}d_{j+1,2k}^{[1]} &= \frac{1}{8}g_{j,k-1}^{[1]} - \frac{1}{8}g_{j,k}^{[1]} + \frac{1}{8}g_{j,k}^{[1]} - \frac{1}{8}g_{j,k+1}^{[1]}\\ &= -\frac{1}{8}\Delta g_{j,k-1}^{[1]} - \frac{1}{8}\Delta g_{j,k}^{[1]}\\ d_{j+1,2k+1}^{[1]} &= \frac{1}{8}\Delta g_{j,k-1}^{[1]} - \frac{3}{8}\Delta g_{j,k}^{[1]}\end{aligned}$$

Hence,

$$||g_{j+1}^{[1]} - g_j^{[1]}||_\infty = ||d_{j+1}^{[1]}||_\infty \leq \frac{1}{2}\max_k |\Delta g_{j,k}^{[1]}|$$

Again, as in the proof of Theorem 1, we can apply Lemma 1 if we can find C' and $\lambda < 1$ such that

$$\max_k |\Delta g^{[1]}_{j,k}| \le C'\lambda^j, \qquad j = 0, 1, 2, \ldots$$

If we take the differences of $g^{[1]}_{j+1,k}$, we get

$$\begin{aligned}
\Delta g^{[1]}_{j+1,2k} &= g^{[1]}_{j+1,2k+1} - g^{[1]}_{j+1,k} && (2.11)\\
&= -\frac{1}{4} g^{[1]}_{j,k-1} + \frac{1}{4} g^{[1]}_{j,k+1} \\
&= \frac{1}{4}\Delta g^{[1]}_{j,k-1} + \frac{1}{4}\Delta g^{[1]}_{j,k} \\
\Delta g^{[1]}_{j+1,2k+1} &= g^{[1]}_{j+1,2k+2} - g^{[1]}_{j+1,k+1} && (2.12)\\
&= \frac{1}{8} g^{[1]}_{j,k-1} - \frac{7}{8} g^{[1]}_{j,k} + \frac{7}{8} g^{[1]}_{j,k+1} - \frac{1}{8} g^{[1]}_{j,k+2} \\
&= -\frac{1}{8}\Delta g^{[1]}_{j,k-1} + \frac{3}{4}\Delta g^{[1]}_{j,k} - \frac{1}{8}\Delta g^{[1]}_{j,k+1}
\end{aligned}$$

From this it follows

$$\max_k |\Delta g^{[1]}_{j+1,k}| \le \max_k |\Delta g^{[1]}_{j,k}|$$

but this only gives

$$\max_k |\Delta g^{[1]}_{j,k}| \le C$$

which is not enough to use Lemma 1. But we can fix this if we use a second iteration of equations (2.11) and (2.12). After some calculations, this gives

$$\begin{aligned}
\Delta g^{[1]}_{j+2,4k} &= -\frac{1}{32}\Delta g^{[1]}_{j,k-2} + \frac{1}{4}\Delta g^{[1]}_{j,k-1} + \frac{1}{8}\Delta g^{[1]}_{j,k} \\
\Delta g^{[1]}_{j+2,4k+1} &= \frac{1}{64}\Delta g^{[1]}_{j,k-2} + \frac{7}{64}\Delta g^{[1]}_{j,k-1} + \frac{7}{64}\Delta g^{[1]}_{j,k} + \frac{1}{64}\Delta g^{[1]}_{j,k+1} \\
\Delta g^{[1]}_{j+2,4k+2} &= \frac{1}{32}\Delta g^{[1]}_{j,k-1} + \frac{1}{4}\Delta g^{[1]}_{j,k} - \frac{1}{32}\Delta g^{[1]}_{j,k+1} \\
\Delta g^{[1]}_{j+2,4k+3} &= -\frac{1}{8}\Delta g^{[1]}_{j,k-1} + \frac{1}{2}\Delta g^{[1]}_{j,k} - \frac{1}{8}\Delta g^{[1]}_{j,k+1}
\end{aligned}$$

From this we get

$$\max_k |\Delta g^{[1]}_{j+2,k}| \le \frac{3}{4} \max_k |\Delta g^{[1]}_{j,k}|$$

We can now apply Lemma 1 with $C = \max_k |\Delta g^{[1]}_{0,k}|/2$ and $\lambda = \sqrt{3/4} < 1$, and part (i) of the proof is complete.

If we move on to part (ii), we observe that as both sides of (2.9) is continuous in x, it is sufficient to show that (ii) holds for all dyadic points

$x = 2^{-J}K$. Then, for all $j \geq J$, we can write $x = 2^{-j}k$, where $k = 2^{j-J}K$. This gives

$$g(x) - g(0) = \sum_{i=0}^{k-1} \big(g_{j,i+1} - g_{j,i}\big) = \sum_{i=0}^{k-1} \big(2^{-j}(i+1) - 2^{-j}i\big) g_{j,i}^{[1]} = A + B$$

where

$$A = \sum_{i=0}^{k-1} \big(2^{-j}(i+1) - 2^{-j}i\big) g^{[1]}(2^{-j}i)$$

and

$$B = \sum_{i=0}^{k-1} \big(2^{-j}(i+1) - 2^{-j}i\big) \big(g_j^{[1]}(2^{-j}i) - g^{[1]}(2^{-j}i)\big)$$

Now, as $j \to \infty$, since $g^{[1]}$ is a continuous function, A converges to the integral in (2.9) and

$$|B| \leq ||g_j^{[1]} - g^{[1]}||_\infty \sum_{i=0}^{k-1} \big(2^{-j}(i+1) - 2^{-j}i\big) = ||g_j^{[1]} - g^{[1]}||_\infty (x - 0) \to 0$$

since $2^{-j}k = x$. This proves (ii), and the proof is complete. □

2.4 Regular, semiregular and irregular subdivision schemes

The way we have defined the four point scheme, the data has to be given uniformly spaced, or regularly. This is not optimal, neither from a design point of view, nor if we want approximate data sampled from a function or measured data. But instead of using the rule in (2.2), we can define $g_{j+1,2k+1}$ using cubic polynomial interpolation. We let $g_{j+1,2k+1}$ be the value at $x_{j+1,2k+1}$ of the unique cubic polynomial that has the value $g_{j,i}$ at $x_{j,i}$ for $i = k-1, k, k+1, k+2$. If the data is regularly spaced, this will be equal to the definition in (2.2).

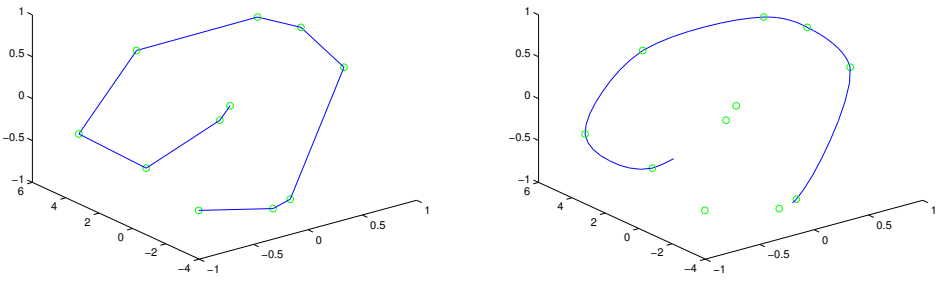
If we let the initial data be spaced nonequally, and let the new points be the midpoint between the old points, we get a semiregular scheme. So, if $x_{0,k}$ are the initial, arbitrary points, then $x_{j+1,2k+1} = (x_{j,k} + x_{j,k+1})/2$ is the new points on level j. The definition of the four point scheme is similar to what we did in the last section, but the proof of smoothness does not work. Joe Warren [11] proved in 1995 that the resulting limit curve is C^1, as the regular scheme.

We can take this even one step further, and let not only the initial data be spaced nonequally, but also choose the new point anywhere in the interval between the old points. This is called an irregular subdivision

scheme. The four point scheme can be adapted to do this as well, but to prove the continuity and smoothness is even more difficult than for the semiregular case. Deubechies et al. [1] introduced the notion of a dyadically balanced grid. If $h_{j,k} = x_{j,k+1} - x_{j,k}$, we then look at

$$\lambda = \sup_{j,k} \max \left(\frac{h_{j+1,2k}}{h_{j,k}}, \frac{h_{j+1,2k+1}}{h_{j,k}} \right)$$

If $1/2 \leq \lambda \leq 1$, the new points are between the old points, and if in addition $\lambda < 1$, the grid is dyadically balanced. It was shown in [1] that a subdivision scheme is C^1 if the scheme is dyadically balanced. It was further shown that the scheme has in fact a derivative that is Holder continuous with exponent $1 - \epsilon$ if $\lambda \leq 2/3$. Floater [6] has later improved this bound, and showed that the subdivision scheme is $C^{2-\epsilon}$ for $\lambda \leq \lambda_0 \approx 0.7142$.



(a) Initial polygon, j=0 (b) After three steps, j=3

Figure 2.2: An initial polygon and the result of three steps of a parametric semiregular scheme using centripetal parameterization.

2.5 Parametric schemes

As already noted, instead of looking at the data as values and grid points, we can look at them as a sequence of points. Now, instead of points just in $\mathbb{R}^2$, we might want to study interpolatory subdivision curves for points in $\mathbb{R}^d$, $d \geq 2$. The simplest way is to find a parametrization of the points, and then use parametric cubic interpolation to calculate new points. So, given a sequence of points $\mathbf{p}_{0,k} \in \mathbb{R}^d$, for each $j \geq 0$, we find parameter values $t_{j,k}$ corresponding to the points $\mathbf{p}_{j,k}$. We set $\mathbf{p}_{j+1,2k} = \mathbf{p}_{j,k}$ and calculate $\mathbf{p}_{j+1,2k+1}$ to be the value of the cubic polynomial interpolating $\mathbf{p}_{j,i}$ at $t_{j,i}$ for $i = k-1, k, k+1, k+2$ at some chosen value t^* in $(t_{j,k}, t_{j,k+1})$.

This rises the question of how to parameterize. One simple solution is to use uniform parameterization, which gives us a parametric version of the standard four point scheme. Another choice is to set the length of each parameter interval to be equal to the euclidian distance between the two corresponding data points, or the square root of this distance. This is called *chordal parameterization* and *centripetal parameterization*. It has been shown that centripetal parameterization with cubic interpolation gives a curve that stays close to the initial polygon, but chordal parameterization gives higher approximation order, see e.g. [7].

If we find a parameterization to the initial points, and instead of calculating a new parametrization for each new j, just insert new parameter values between the old, so for each $j > 0$ let $t_{j+1,2k} = t_{j,k}$ and $t_{j+1,2k+1} \in (t_{j,k}, t_{j,k+1})$, we get back to the irregular schemes of the previous section. So, according to Deubechies et al. [1] all parametric subdivision schemes defined this way are C^1.

The non-linear scheme studied by Dyn, Floater and Hormann [3] used the other possibility, to calculate a new parameterization for each step j. They defined a parameterization by $t_{j,0} = 0$ and $t_{j,k+1} = t_{j,k} + ||\mathbf{p}_{j,k+1} - \mathbf{p}_{j,k}||^\alpha$, where $\alpha = 1$ gives chordal parameterization, $\alpha = 1/2$ gives centripetal and $\alpha = 0$ gives uniform. Now, they used cubic interpolation, and set $\mathbf{p}_{j+1,2k+1}$ to the value of the interpolating polynomial at $(t_{j,k} + t_{j,k+1})/2$. Since parametric interpolation is only defined when the parameter values are distinct, this scheme relies on the points to be distinct. If any two initial neighbour points $\mathbf{p}_{0,k}$ and $\mathbf{p}_{0,k+1}$ are distinct, $\mathbf{p}_{0,k} \neq \mathbf{p}_{0,k+1}$, Dyn et al. showed that any two consecutive points at level j are distinct.

This scheme is non-linear, because the new parameterization can not be expressed as a linear combination of the previous parameterization. It was proved to be continuous by bounds on the distance between the midpoints of two consecutive points on level j and the new points on level $j+1$. This gives bounds to the distance between the points on level $j+1$. This iterated scheme is not proved be be C^1, but numerical examples suggests it is.

Chapter 3

Tensor product surface interpolation

3.1 Tensor product spline interpolation

In the previous chapter we discussed interpolating methods for curves. In this and the next chapter we will discuss interpolating methods for surfaces.

We start by defining tensor product splines, let $\{\phi_i\}_{i=1}^{n_1}$ be the B-splines of degree d on a knot vector $\boldsymbol{\tau}$, and $\{\psi_j\}_{j=1}^{n_2}$ be the B-splines of degree l on the knots $\boldsymbol{\sigma}$.

Definition 3. *The tensor product of the two spline spaces* $\mathbb{S}_1 = \text{span}\{\phi_1, \ldots, \phi_{n_1}\}$ *and* $\mathbb{S}_2 = \text{span}\{\psi_1, \ldots, \psi_{n_2}\}$ *is defined to be the family of all functions of the form*

$$g(x, y) = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} c_{i,j} \phi_i(x) \psi_j(y)$$

where the coefficients $(c_{i,j})_{i,j=1}^{n_1,n_2}$ *can be any real numbers. This linear space of functions is denoted* $\mathbb{S}_1 \otimes \mathbb{S}_2$.

If we now move on to interpolation, we are given a set of gridded data

$$(x_i, y_j, f_{i,j})_{i=1,j=1}^{m_1,m_2},$$

where $x_1 < x_2 < \cdots < x_{m_1}$ and $y_1 < y_2 < \cdots < y_{m_2}$. We want to find a function $g = g(x, y)$ in a tensor product spline space $\mathbb{S}_1 \otimes \mathbb{S}_2$ such that

$$g(x_i, y_j) = f_{i,j}, \qquad i = 1, \ldots, m_1, \qquad j = 1, \ldots, m_2.$$

Here $\mathbb{S}_1$ and $\mathbb{S}_2$ are two univariate spline spaces $\mathbb{S}_1 = \text{span}\{\phi_1, \ldots, \phi_{m_1}\}$ and $\mathbb{S}_2 = \text{span}\{\psi_1, \ldots, \psi_{m_2}\}$, where $(\phi_i)_{i=1}^{m_1}$ are the basis of the B-splines

for the spline space $\mathbb{S}_1$, and similary $(\psi_i)_{i=1}^{m_2}$ are the B-spline basis of $\mathbb{S}_2$. We can now write g as

$$g(x,y) = \sum_{p=1}^{m_1}\sum_{q=1}^{m_2} c_{p,q}\phi_p(x)\psi_q(y) \tag{3.1}$$

which, with the interpolation conditions, gives a set of equations

$$\sum_{p=1}^{m_1}\sum_{q=1}^{m_2} c_{p,q}\phi_p(x_i)\psi_q(y_j) = f_{i,j}, \qquad \forall i,j.$$

This double sum can be split into two sets of simple sums

$$\sum_{p=1}^{m_1} d_{p,j}\phi_p(x_i) = f_{i,j}, \tag{3.2}$$

$$\sum_{q=1}^{m_2} c_{p,q}\psi_q(y_j) = d_{p,j} \tag{3.3}$$

We now define the matrices

$$\boldsymbol{\Phi} = (\phi_p(x_i)), \qquad \boldsymbol{\Psi} = (\psi_q(y_j))$$
$$\boldsymbol{D} = (d_{p,j}), \qquad \boldsymbol{F} = (f_{i,j}), \qquad \boldsymbol{C} = (c_{p,q})$$

We observe that the simple sums in (3.2) and (3.3) now can be written as

$$(\boldsymbol{\Phi D})_{i,j} = (\boldsymbol{F})_{i,j}$$
$$(\boldsymbol{\Psi C}^T)_{j,p} = (\boldsymbol{D}^T)_{j,p}$$

which, in matrix form, becomes

$$\boldsymbol{\Phi D} = \boldsymbol{F} \qquad \text{and} \qquad \boldsymbol{C\Psi}^T = \boldsymbol{D}$$

Lemma 3. *Suppose the matrices $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$ are nonsingular. Then there is a unique $g \in \mathbb{S}_1 \otimes \mathbb{S}_2$ such that the interpolation property $g(x_i, y_j) = f_{i,j}$ holds for all i and j. This g is a tensor product spline, and the coefficient matrix $\boldsymbol{C} = (c_{p,q})$ fulfills*

$$\boldsymbol{\Phi C\Psi}^T = \boldsymbol{F}.$$

Now, if we give some boundary conditions, and let $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$ be as discussed in section 2.1.1, we would get a C^2 tensor product cubic spline interpolant. The smoothness property follows from the definition of g. If we differentiate g with respect either to x or y, we see from (3.1) that we only differentiate functions of $\mathbb{S}_1$ or $\mathbb{S}_2$, respectively, and that functions of these spaces are C^2.

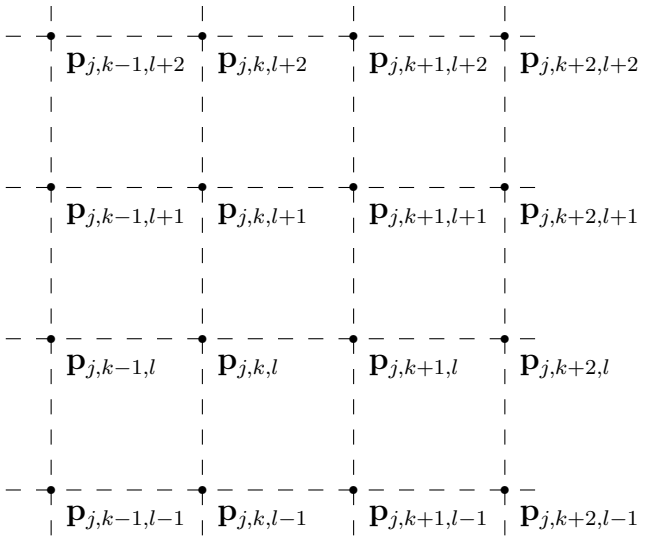


Figure 3.1: The structure of the points in a surface subdivision scheme.

3.2 Tensor product subdivision scheme

If we are given points $\mathbf{p}_{k,l} \in \mathbb{R}^d$, $k, l \in \mathbb{Z}$, $d \geq 3$, in a quadrilateral mesh, we can use a tensor product interpolating subdivision scheme to generate a smooth surface. We use one of the subdivision schemes for curves first in one direction, then the other. Because of the tensor product structure, we need the parameterization to be the same for all curves in one direction, so we can only have two different parameterizations. The simplest choice is then to make use of one, uniform parameterization. This then becomes the tensor product of the four point scheme.

We set $\mathbf{p}_{0,k,l} = \mathbf{p}_{k,l}$. Then for all $j \geq 0$, we set $\mathbf{p}_{j+1,2k,2l} = \mathbf{p}_{j,k,l}$ for all k, l. We are then left with creating the new points. The *edge points*, $\mathbf{p}_{j+1,2k+1,2l}$ and $\mathbf{p}_{j+1,2k,2l+1}$ are created using the standard four point rule:

$$\mathbf{p}_{j+1,2k+1,2l} = -\frac{1}{16}\mathbf{p}_{j,k-1,l} + \frac{9}{16}\mathbf{p}_{j,k,l} + \frac{9}{16}\mathbf{p}_{j,k+1,l} - \frac{1}{16}\mathbf{p}_{j,k+2,l}$$
$$\mathbf{p}_{j+1,2k,2l+1} = -\frac{1}{16}\mathbf{p}_{j,k,l-1} + \frac{9}{16}\mathbf{p}_{j,k,l} + \frac{9}{16}\mathbf{p}_{j,k,l+1} - \frac{1}{16}\mathbf{p}_{j,k,l+2}$$

We are now left with the *face point*, $\mathbf{p}_{j+1,2k+1,2l+1}$. For this we can use the curve scheme at the new edge points in either direction, we can use the four point scheme on $\mathbf{p}_{j+1,2k+1,2l-2}$, $\mathbf{p}_{j+1,2k+1,2l}$, $\mathbf{p}_{j+1,2k+1,2l+2}$ and $\mathbf{p}_{j+1,2k+1,2l+4}$, or on the points $\mathbf{p}_{j+1,2k-2,2l+1}$, $\mathbf{p}_{j+1,2k,2l+1}$, $\mathbf{p}_{j+1,2k+2,2l+1}$ and $\mathbf{p}_{j+1,2k+4,2l+1}$. But if we do the calculations these two give the same result,

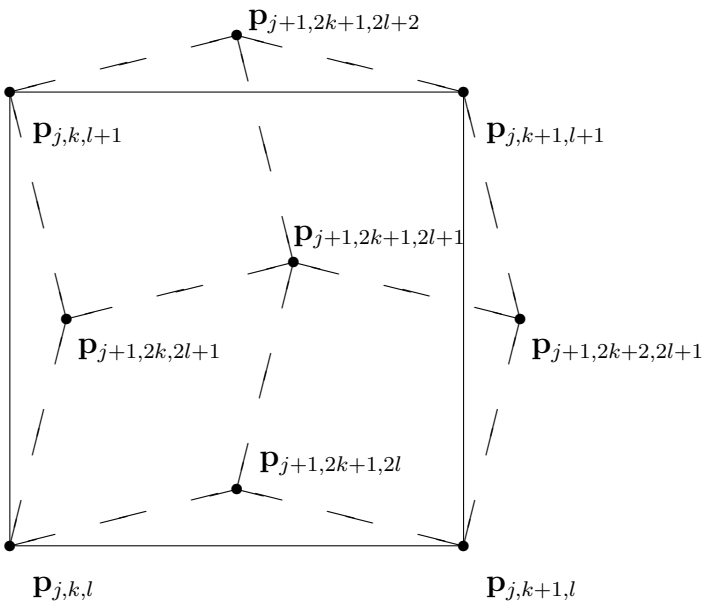


Figure 3.2: New points for a subdivision surface scheme at level $j+1$.

and is the same as calculating the face point as

$$\begin{aligned}\mathbf{p}_{j+1,2k+1,2l+1} = \frac{1}{256}\Big(& \mathbf{p}_{j,k-1,l-1} - 9\mathbf{p}_{j,k-1,l} - 9\mathbf{p}_{j,k-1,l+1} + \mathbf{p}_{j,k-1,l+2} \\ & -9\mathbf{p}_{j,k,l-1} + 81\mathbf{p}_{j,k,l} + 81\mathbf{p}_{j,k,l+1} - 9\mathbf{p}_{j,k,l+2} \\ & -9\mathbf{p}_{j,k+1,l-1} + 81\mathbf{p}_{j,k+1,l} + 81\mathbf{p}_{j,k+1,l+1} - 9\mathbf{p}_{j,k+1,l+2} \\ & +\mathbf{p}_{j,k+2,l-1} - 9\mathbf{p}_{j,k+2,l} - 9\mathbf{p}_{j,k+2,l+1} + \mathbf{p}_{j,k+2,l+2}\Big)\end{aligned}$$

3.2.1 Convergence and smoothness

The four point subdivision scheme for curves is a linear scheme, e.g. the data points on level $j+1$ can be expressed as a linear combination of the data points on level j. This means that we can write the limit of the four point scheme as

$$g(x) = \sum_k f_k \phi_k(x)$$

where f_k are the initial data, and $\phi_k(x)$ is the basis function of the four point scheme, it is the limit of the scheme applied to $s_k = 1$ and $s_t = 0$ for all $t \neq k$. As $\phi_k(x)$ is a limit of the scheme, we know that it exist, and is $C^{2-\epsilon}$.

Since the uniform tensor product interpolatory scheme is based on taking the four point scheme first in one direction, then the other, the limit surface can be written as

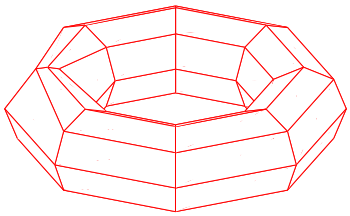
$$\mathbf{g}(x,y) = \sum_{k,l} \mathbf{p}_{k,l}\phi_k(x)\phi_l(y)$$

If we want to differentiate $\mathbf{g}$ with respect to either x or y, we would only differentiate $\phi_k(x)$ or $\phi_l(y)$, which both are $C^{2-\epsilon}$. Hence the limit of the tensor product subdivision scheme is $C^{2-\epsilon}$.

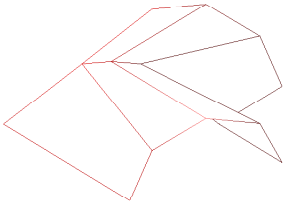
3.3 Problems with the tensor product approach

As mentioned, with the tensor product comes certain problems, namely the limitations of parameterization. This means that we can only have one parameterization in each direction. The way we usually find such a parameterization, is to find the best parameter values for each row and column of the data, and then preform some averaging process for each direction. But this only gives satisfactory results if the data yields similar parameterizations.

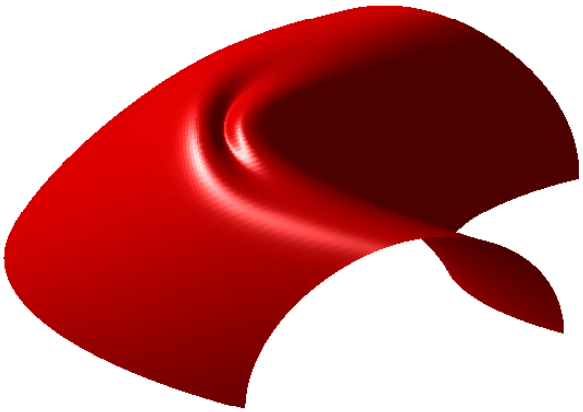
If we look at figure 3.3 we see the problem. In the torus showed in figure 3.3(a) we see that for seven of the eight columns going around the torus, the uniform parameterization is the best choice. Hence, if we average the best parametrization for each column, the result is nearly uniform. This, however means that the surface has unexpected behavior near the last column. Figure 3.3(b) shows the nonuniform part of the last column, and figure 3.3(c) and figure 3.3(d) shows the result of using tensor product splines and tensor product subdivision on the initial polygon, where the result is restricted to the polygon in figure 3.3(b).



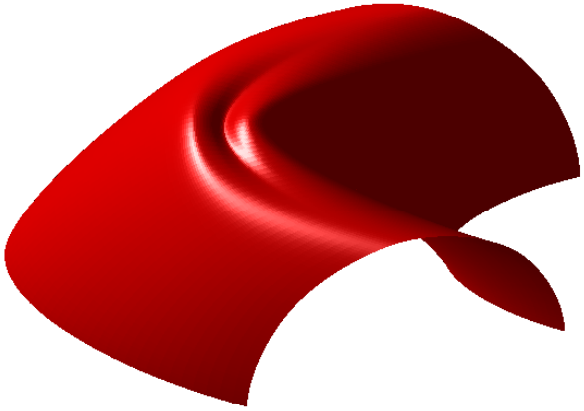
(a) Initial polygon.



(b) An interesting part of the initial polygon.



(c) The result of tensor product spline interpolation on the interesting part.



(d) The result of tensor product four point subdivision on the interesting part.

Figure 3.3: The problem with the tensor product approach.

Chapter 4

New methods for surface interpolation

In this chapter we will propose two new methods for interpolating surfaces. Both will solve the parameterization problems of the tensor product methods.

4.1 A 16-point subdivision scheme

If we are given points $\mathbf{p}_{k,l} \in \mathbb{R}^d$ in a quadrilateral mesh, where $k, l \in \mathbb{Z}$, we want to calculate a surface passing through these points. We assume that the points are distinct from their neighbours, $\mathbf{p}_{k,l} \neq \mathbf{p}_{k+1,l}$ and $\mathbf{p}_{k,l} \neq \mathbf{p}_{k,l+1}$ for all k, l.

We now want to find a parameterization of the points. We will use centripetal parameterization and calculate a mesh in two dimensions of parameter values. We start by setting $u_{0,l} = 0$ for $l \in \mathbb{Z}$, and then let

$$u_{i,l} = u_{i-1,l} + ||\mathbf{p}_{i,l} - \mathbf{p}_{i-1,l}||^{\frac{1}{2}} \qquad \forall i, l \in \mathbb{Z}$$

Similarly, we set $v_{k,0} = 0$, and then calculate

$$v_{k,i} = v_{k,i-1} + ||\mathbf{p}_{k,i} - \mathbf{p}_{k,i-1}||^{\frac{1}{2}} \qquad \forall i, k \in \mathbb{Z}$$

We now set $\mathbf{t}_{k,l} = (u_{k,l}, v_{k,l})$, and we have our proposed mesh.

Now, to find an interpolating surface, we will use semiregular subdivision. So, first we set $\mathbf{p}_{0,k,l} = \mathbf{p}_{k,l}$ and $\mathbf{t}_{0,k,l} = \mathbf{t}_{k,l}$. Now, for each step, we want to update our parameterization and find new points. Since we use a semiregular approach, finding a new parameterization is quite simple. For each $j \geq 0$, we set $\mathbf{t}_{j+1,2k,2l} = \mathbf{t}_{j,k,l}$ to keep the old parameter values, and

then find the new values by

$$\begin{aligned}
\mathbf{t}_{j+1,2k+1,2l} &= \frac{\mathbf{t}_{j,k,l} + \mathbf{t}_{j,k+1,l}}{2} \\
\mathbf{t}_{j+1,2k,2l+1} &= \frac{\mathbf{t}_{j,k,l} + \mathbf{t}_{j,k,l+1}}{2} \\
\mathbf{t}_{j+1,2k+1,2l+1} &= \frac{\mathbf{t}_{j,k,l} + \mathbf{t}_{j,k+1,l} + \mathbf{t}_{j,k,l+1} + \mathbf{t}_{j,k+1,l+1}}{4}
\end{aligned}$$

Varignon's theorem states that the midpoints of the sides of any arbitrary quadrilateral forms a parallelogram. Using this, and the property that the diagonals of a parallelogram bisect, $\mathbf{t}_{j+1,2k+1,2l+1}$ is in fact the midpoint on the line both through $\mathbf{t}_{j+1,2k+1,2l}$ and $\mathbf{t}_{j+1,2k+1,2l+2}$, and the line through $\mathbf{t}_{j+1,2k,2l+1}$ and $\mathbf{t}_{j+1,2k+2,2l+1}$.

Let $F[t_0, \ldots, t_n; \mathbf{p}_0, \ldots, \mathbf{p}_n](t)$ be the unique polynomial of degree less than or equal to n that interpolates the points $\mathbf{p}_0$ to $\mathbf{p}_n$ at parameter values t_0 to t_n, respectively. Now, for creating new points, we set $\mathbf{p}_{j+1,2k,2l} = \mathbf{p}_{j,k,l}$ to get the interpolation property. The method is then to first use parametric curve subdivision along the rows, then along both the new and old columns. So, first, for all k, l, do

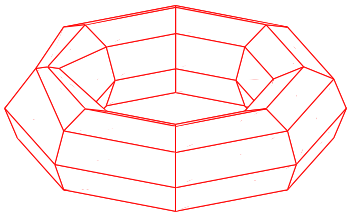
$$\begin{aligned}
\mathbf{p}_{j+1,2k+1,2l} =& F[u_{j,k-1,l}, u_{j,k,l}, u_{j,k+1,l}, u_{j,k+2,l}; \\
& \mathbf{p}_{j,k-1,l}, \mathbf{p}_{j,k,l}, \mathbf{p}_{j,k+1,l}, \mathbf{p}_{j,k+2,l}](u_{j+1,2k+1,2l})
\end{aligned}$$

then, for all k, l, do

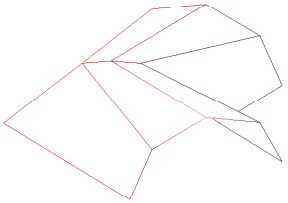
$$\begin{aligned}
\mathbf{p}_{j+1,k,2l+1} =& F[v_{j+1,k,2l-2}, v_{j+1,k,2l}, v_{j+1,k,2l+2}, v_{j+1,k,2l+4}; \\
& \mathbf{p}_{j+1,k,2l-2}, \mathbf{p}_{j+1,k,2l}, \mathbf{p}_{j+1,k,2l+2}, \mathbf{p}_{j+1,k,2l+4}](v_{j+1,k,2l+1})
\end{aligned}$$

This scheme is not symmetric, but by visual inspection, there seems to be little difference in doing the rows first or the columns first. Studying numerical examples, the maximum distance between the polygons created by doing rows first or columns first is in the area of $1 - 2\%$ of the distance between the points in the initial polygon.

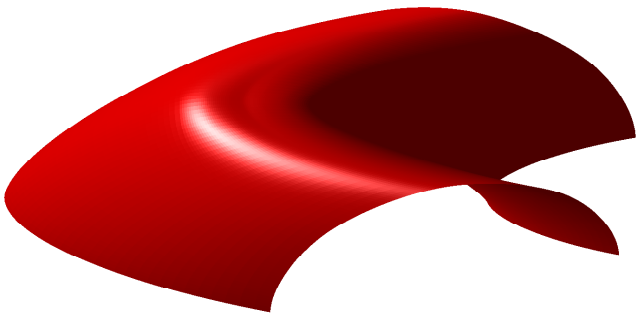
If we insist on using a symmetric method, we can do this easily by, for the face point, using the average of the curve method in the rows and



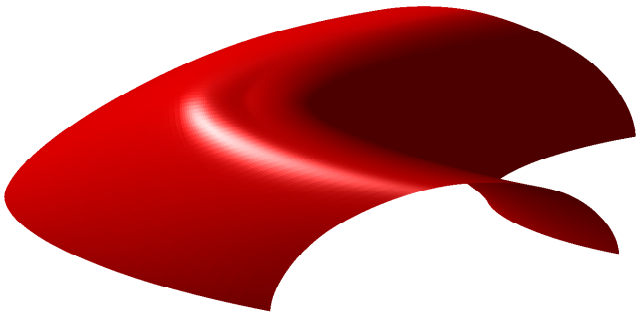
(a) Initial polygon.



(b) An interesting part of the initial polygon.



(c) The result of the unsymmetric method on the interesting part.



(d) The result of symmetric method on the interesting part.

Figure 4.1: A numerical example with the same initial data as in figure 3.3.

columns. This gives

$$
\begin{aligned}
\mathbf{p}_{j+1,2k+1,2l} =& F[u_{j,k-1,l}, u_{j,k,l}, u_{j,k+1,l}, u_{j,k+2,l};\\
&\mathbf{p}_{j,k-1,l}, \mathbf{p}_{j,k,l}, \mathbf{p}_{j,k+1,l}, \mathbf{p}_{j,k+2,l}](u_{j+1,2k+1,2l}) \qquad \forall k, l\\
\mathbf{p}_{j+1,2k,2l+1} =& F[v_{j,k,l-1}, v_{j,k,l}, v_{j,k,l+1}, v_{j,k,l+2};\\
&\mathbf{p}_{j,k,l-1}, \mathbf{p}_{j,k,l}, \mathbf{p}_{j,k,l+1}, \mathbf{p}_{j,k,l+2}](v_{j+1,2k,2l+1}) \qquad \forall k, l\\
\mathbf{p}_{j+1,2k+1,2l+1} =& \frac{1}{2}\Big(F[u_{j+1,2k-2,2l+1}, u_{j+1,2k,2l+1}, u_{j+1,2k+2,2l+1}, u_{j+1,2k+4,2l+1};\\
&\mathbf{p}_{j+1,2k-2,2l+1}, \mathbf{p}_{j+1,2k,2l+1}, \mathbf{p}_{j+1,2k+2,2l+1}, \mathbf{p}_{j+1,2k+4,2l+1}]\\
&(u_{j+1,2k+1,2l+1})\\
&+ F[v_{j+1,2k+1,2l-2}, v_{j+1,2k+1,2l}, v_{j+1,2k+1,2l+2}, v_{j+1,2k+1,2l+4};\\
&\mathbf{p}_{j+1,2k+1,2l-2}, \mathbf{p}_{j+1,2k+1,2l}, \mathbf{p}_{j+1,2k+2,2l+2}, \mathbf{p}_{j+1,2k+1,2l+2}]\\
&(v_{j+1,2k+1,2l+1})\Big) \qquad \forall k, l
\end{aligned}
$$

In figure 4.1 we see the result of these new proposed methods applied to the same data as in figure 3.3, and the result is visually pleasing, and closer to what you would expect with respect to the initial data than the tensor product methods. If we do the unsymmetric method with the rows first, or the columns first, we still, after a number of steps, get the same number of points, in the same structure. Hence we can look at the difference in distance between corresponding points from these two. The largest such difference between the two unsymmetric methods appear as the first face point inserted in one of the enlarged quadrilaterals on the inside of the torus, in fact in the quadrilateral which appear in the top right of figure 4.1(b). The difference is about 2% of the length of the shortest side of this quadrilateral and about 1% of the longest. In figure 4.2 the result of the unsymmetric methods with rows first and columns first are plotted in red and blue in the same figure.

4.1.1 Smoothness

To discuss the continuity and smoothness of this scheme, we need to define what surface we are studying. This definition is a generalization of the limit of the polygons we used in chapter 2.

We let $\mathbf{g}_j(r, s)$ be the parametric bilinear interpolant to $(\mathbf{p}_{j,k,l}, \mathbf{t}_{j,k,l})$.

Figure 4.2: The two unsymmetric methods plotted together, one in blue and one in red, with the initial data as in 4.1.

This means that if $r \in [u_{j,k,l}, u_{j,k+1,l}]$ and $s \in [v_{j,k,l}, v_{j,k,l+1}]$, then

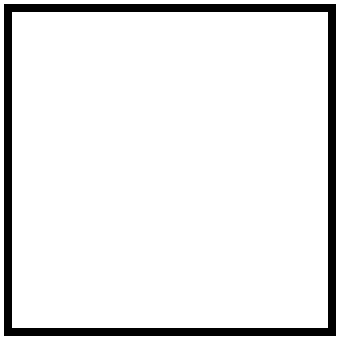
$$\begin{aligned}\mathbf{g}_j(r,s) = &\frac{u_{j,k+1,l}-r}{u_{j,k+1,l}-u_{j,k,l}}\frac{v_{j,k,l+1}-s}{v_{j,k,l+1}-v_{j,k,l}}\mathbf{p}_{j,k,l}\\ &+\frac{r-u_{j,k,l}}{u_{j,k+1,l}-u_{j,k,l}}\frac{v_{j,k,l+1}-s}{v_{j,k,l+1}-v_{j,k,l}}\mathbf{p}_{j,k+1,l}\\ &+\frac{u_{j,k+1,l}-r}{u_{j,k+1,l}-u_{j,k,l}}\frac{s-v_{j,k,l}}{v_{j,k,l+1}-v_{j,k,l}}\mathbf{p}_{j,k,l+1}\\ &+\frac{r-u_{j,k,l}}{u_{j,k+1,l}-u_{j,k,l}}\frac{s-v_{j,k,l}}{v_{j,k,l+1}-v_{j,k,l}}\mathbf{p}_{j,k+1,l+1}\end{aligned}$$

We can now look at the sequence $\mathbf{g}_0, \mathbf{g}_1, \mathbf{g}_2, \ldots$, and the limit of this sequence,

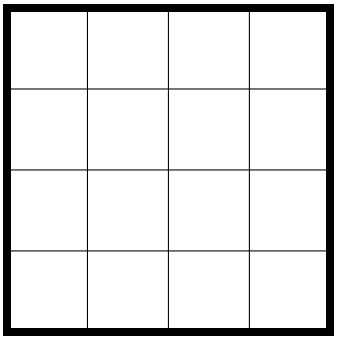
$$\mathbf{g} = \lim_{j\to\infty} \mathbf{g}_j$$

I have not been able to prove that this limit exist, and hence neither the continuity nor the smoothness of the limit surface of this scheme. The main problem is to generalize the (rather hard) proofs of continuity of semiregular curve subdivision schemes to surfaces. The most promising method seems to be to generalize the piecewise polynomial approach used by Floater in [6], but I was not able to succeed with this. However, numerical results, as we will study in the next chapter, seems to confirm that this method is in fact C^1. In this section we will study some results which supports this claim.

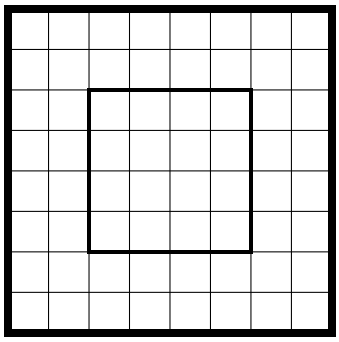
If the data we are interpolating has a special structure such that the parameterizations for all the columns are equal, and also equal for all the rows, this method is C^1. This follows from the fact that this method then reduces to the tensor product of a semiregular subdivision scheme.



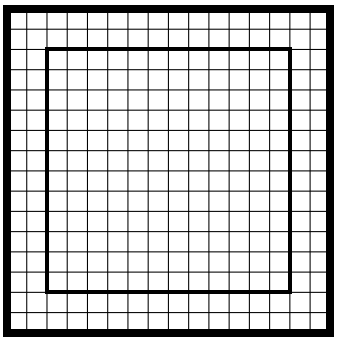
(a) Initial quadrilateral.



(b) The quadrilateral after two steps.



(c) The quadrilateral after three steps, where the inner bold lines marks for which part we use tensor product in any further steps.



(d) The quadrilateral after four steps, where the inner bold lines marks for which part we use tensor product in any further steps.

Figure 4.3: The growth of the tensor product patch in an initial quadrilateral.

Further, we know that if we look at a curve at the resulting limit surface along any row or column of points, this curve is C^1. This follows from the definition of the surface scheme, along each row and column we only use a semiregular curve subdivison scheme, which we know is C^1. So, according to this, for any j we have a bilinear interpolant, for which we can, using this scheme, find limit curves along the rows and columns of interpolation points that are C^1.

Another support of our C^1 claim follows from the semiregularity of the scheme. After a few steps, in the interior of each of the initial quadrilaterals, we have a parameterization which for each row and column is a uniform parameterization. Then, for the calculation of new points which only uses interior points, this reduces to the tensor product. And for each step, the region inside the initial quadrilateral which we use the tensor product method will expand towards the edges of the quadrilateral.

If we look at figure 4.3 we can study this statement more closely. In figure 4.3(c) we see the result of three steps with the surface subdivision scheme. The bold border marks the edges of the initial polygon, while the bold inner quadrilateral marks the part for which we will for all further steps use tensor product uniform subdivision. Similarly, in figure 4.3(d) we see the result of four steps, and for which part we will use tensor product for further steps. As this figure shows, from the third iteration of the subdivision scheme, we get a patch in each initial quadrilateral where we will use a tensor product method, and hence the limit surface over this patch is C^1.

From figure 4.3 is seems as the distance between the border of the quadrilateral and the patch which uses tensor product is halved for each step, but this is not the case. However, for the parameter values this is true.

4.2 A 12-point interpolatory surface scheme

During an attempt to prove the continuity of the 16-point subdivision scheme discussed in the last section, I stumbled upon an idea for a 12-point surface scheme. This scheme seems to behave quite similarly to the 16-point scheme, but uses less data.

The 16-point scheme studied in section 4.1 is called so because it uses all of the 16 points in figure 3.1 to calculate the new points in figure 3.2. In fact, it is only for the face point all the 16 points are used, calculations of the new edge points does not use the corner points in figure 3.1, but the face point uses four (or eight for the symmetric) new edge points, which each does need four points. The 12-point scheme described next does not use the corner points in the calculation of the new face point.

We start similar to section 4.1, we are given points $\mathbf{p}_{k,l} \in \mathbb{R}^d$ in a quadrilateral mesh, where $k, l \in \mathbb{Z}$, and we assume that the points are distinct from its neighbours, $\mathbf{p}_{k,l} \neq \mathbf{p}_{k+1,l}$ and $\mathbf{p}_{k,l} \neq \mathbf{p}_{k,l+1}$ for all k, l. We find and update the parameterization in exactly the same way as for the 16-point scheme, so for each $j \geq 0$ we have a mesh of parameters $\mathbf{t}_{j,k,l} = (u_{j,k,l}, v_{j,k,l})$. Now, for each step, we keep the old points, for the edge points use the same rules as for the symmetric 16-point scheme, and use a rule based on the bilinearly blended coons patch for the face point.

Coons patches were invented by S. Coon, who was a consultant for Ford. We are given four arbitrary curves $s_1(u), s_2(u), r_1(v)$ and $r_2(v)$ where $u, v \in [0, 1]$. We now want to find a surface g which has these four curves as boundary curves such that

$$\begin{aligned} g(u,0) &= s_1(u) & g(u,1) &= s_2(u) \\ g(0,v) &= r_1(v) & g(1,v) &= r_2(v) \end{aligned}$$

These four curves define two ruled surfaces

$$l_s(u,v) = (1-v)s_1(u) + vs_2(u)$$

and

$$l_r(u,v) = (1-u)r_1(v) + ur_2(v)$$

Both of these surfaces interpolate two of the boundary curves, but fail on the other two, where they are linear. So if we sum them, and subtract the bilinear interpolant l_{sr} to the four corners, we get a surface of the form we want. So the billinearly blended Coons patch is

$$g = l_s + l_r - l_{sr}$$

where

$$l_{sr}(u,v) = \begin{bmatrix} 1-u & u \end{bmatrix} \begin{bmatrix} g(0,0) & g(0,1) \\ g(1,0) & g(1,1) \end{bmatrix} \begin{bmatrix} 1-v \\ v \end{bmatrix}$$

A more comprehensive study of Coons patches can be found in chapter 22 of [5].

For the 12-point method the four boundary curves are given by the interpolating polynomial we use to find the edge points, between the parameter values for the corners. These parameter values are very rarely 0 and 1, so we have to do some translation. This is pretty straight forward, but because of the semiregular approach we use, we are only interested in $u = v = 1/2$. This means that we only want

$$g(1/2,1/2) = l_s(1/2,1/2) + l_r(1/2,1/2) - l_{sr}(1/2,1/2)$$

Now, since the midpoints of the boundary curves are the new edge points, we get

$$l_s(1/2, 1/2) = (\mathbf{p}_{j+1,2k+1,2l} + \mathbf{p}_{j+1,2k+1,2l+2})/2$$

and

$$l_r(1/2, 1/2) = (\mathbf{p}_{j+1,2k,2l+1} + \mathbf{p}_{j+1,2k+2,2l+1})/2$$

The corner points of the boundary curves are the old points, so

$$l_{sr}(1/2, 1/2) = (\mathbf{p}_{j,k,l} + \mathbf{p}_{j,k+1,l} + \mathbf{p}_{j,k,l+1} + \mathbf{p}_{j,k+1,l+1})/4$$

Combining these we get the following rules for updating one iteration of the 12-point scheme: First, for all k, l, do

$$\begin{aligned}
\mathbf{p}_{j+1,2k,2l} &= \mathbf{p}_{j,k,l} \\
\mathbf{p}_{j+1,2k+1,2l} &= F[u_{j,k-1,l}, u_{j,k,l}, u_{j,k+1,l}, u_{j,k+2,l}; \\
&\qquad \mathbf{p}_{j,k-1,l}, \mathbf{p}_{j,k,l}, \mathbf{p}_{j,k+1,l}, \mathbf{p}_{j,k+2,l}](u_{j+1,2k+1,2l}) \\
\mathbf{p}_{j+1,2k,2l+1} &= F[v_{j,k,l-1}, v_{j,k,l}, v_{j,k,l+1}, v_{j,k,l+2}; \\
&\qquad \mathbf{p}_{j,k,l-1}, \mathbf{p}_{j,k,l}, \mathbf{p}_{j,k,l+1}, \mathbf{p}_{j,k,l+2}](v_{j+1,2k,2l+1})
\end{aligned}$$

Then, for the face points, do, for all k, l

$$\begin{aligned}
\mathbf{p}_{j+1,2k+1,2l+1} = &\frac{\mathbf{p}_{j+1,2k+1,2l} + \mathbf{p}_{j+1,2k,2l+1} + \mathbf{p}_{j+1,2k+1,2l+2} + \mathbf{p}_{j+1,2k+2,2l+1}}{2} \\
&- \frac{\mathbf{p}_{j,k,l} + \mathbf{p}_{j,k+1,l} + \mathbf{p}_{j,k,l+1} + \mathbf{p}_{j,k+1,l+1}}{4}
\end{aligned}$$

Here $F[t_0, \ldots, t_n; \mathbf{p}_0, \ldots, \mathbf{p}_n](t)$ is as in section 4.1, it is the unique polynomial of degree less than or equal to n that interpolates the points $\mathbf{p}_0$ to $\mathbf{p}_n$ at parameter values t_0 to t_n, respectively.

4.2.1 Smoothness

Like the 16-point scheme, we have not been able to prove convergence of the scheme in the general case. However, in the regular case with uniform parametrization, there are some interesting observations. In this case, the scheme becomes

$$\begin{aligned}
\mathbf{p}_{j+1,2k,2l} &= \mathbf{p}_{j,k,l} \\
\mathbf{p}_{j+1,2k+1,2l} &= -\frac{1}{16}\mathbf{p}_{j,k-1,l} + \frac{9}{16}\mathbf{p}_{j,k,l} + \frac{9}{16}\mathbf{p}_{j,k+1,l} - \frac{1}{16}\mathbf{p}_{j,k+2,l} \\
\mathbf{p}_{j+1,2k,2l+1} &= -\frac{1}{16}\mathbf{p}_{j,k,l-1} + \frac{9}{16}\mathbf{p}_{j,k,l} + \frac{9}{16}\mathbf{p}_{j,k,l+1} - \frac{1}{16}\mathbf{p}_{j,k,l+2}
\end{aligned}$$

and, for the face point

$$\begin{aligned}
\mathbf{p}_{j+1,2k+1,2l+1} = &\frac{\mathbf{p}_{j+1,2k+1,2l} + \mathbf{p}_{j+1,2k,2l+1} + \mathbf{p}_{j+1,2k+1,2l+2} + \mathbf{p}_{j+1,2k+2,2l+1}}{2} \\
&- \frac{\mathbf{p}_{j,k,l} + \mathbf{p}_{j,k+1,l} + \mathbf{p}_{j,k,l+1} + \mathbf{p}_{j,k+1,l+1}}{4}
\end{aligned}$$

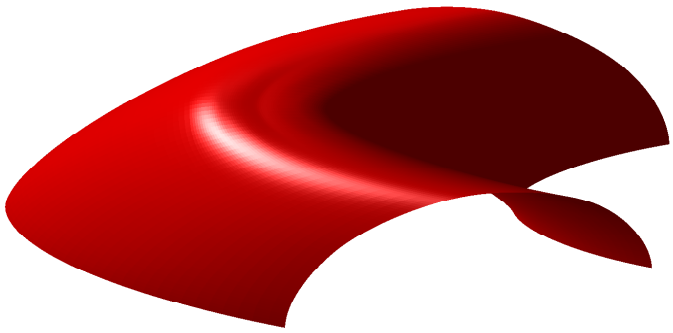


Figure 4.4: A numerical example of the 12-point scheme with the same initial data as in Figure 4.1.

Here, we have explicit rules for $\mathbf{p}_{j+1,2k+1,2l}$, $\mathbf{p}_{j+1,2k,2l+1}$, $\mathbf{p}_{j+1,2k+1,2l+2}$ and $\mathbf{p}_{j+1,2k+2,2l+1}$. If we insert these rules, we get, after some calulations

$$\begin{aligned}\mathbf{p}_{j+1,2k+1,2l+1} = & \quad -\frac{1}{32}\mathbf{p}_{j,k,l-1} - \frac{1}{32}\mathbf{p}_{j,k+1,l-1} \\ & -\frac{1}{32}\mathbf{p}_{j,k-1,l} + \frac{5}{16}\mathbf{p}_{j,k,l} + \frac{5}{16}\mathbf{p}_{j,k+1,l} - \frac{1}{32}\mathbf{p}_{j,k+2,l} \\ & -\frac{1}{32}\mathbf{p}_{j,k-1,l+1} + \frac{5}{16}\mathbf{p}_{j,k,l+1} + \frac{5}{16}\mathbf{p}_{j,k+1,l+1} - \frac{1}{32}\mathbf{p}_{j,k+2,l+1} \\ & \quad -\frac{1}{32}\mathbf{p}_{j,k,l+2} - \frac{1}{32}\mathbf{p}_{j,k+1,l+2}\end{aligned}$$

This is the same as the tensor-product-like scheme with $w = 1/16$ studied by Sigalit Hed in her master thesis [9]. She proves that this scheme is C^1 in the regular case.

From this we can make the same arguments as in section 4.1.1, so also the 16-point scheme has smooth limit curves and smooth patches in each initial quadrilateral.

4.3 A non-linear scheme

Our first idea was to generalize the non-linear scheme based on iterated parameterization by Dyn, Floater and Hormann [3]. A surface scheme based on this method would be very similar to the 16-point scheme described in section 4.1, the calculation of new point would be identical, but a new parameterization for each step would be calculated using the method used for finding $\mathbf{t}_{0,k,l}$ for the semi-regular 16-point scheme.

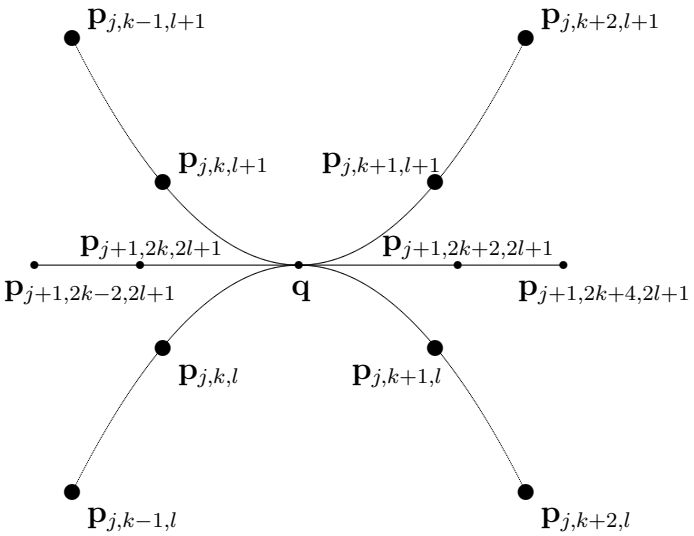


Figure 4.5: A sketch of how subdivision can create new, equal points. In the figure, $\mathbf{q}$ is equal to $\mathbf{p}_{j+1,2k+1,2l}$, $\mathbf{p}_{j+1,2k+1,2l+1}$ and $\mathbf{p}_{j+1,2k+1,2l+2}$.

The main problem with this approach, is that it might not be well defined. To calculate a parameterization of the points, we need any adjoining points to be distinct. In the case of curves, it was shown that if each pair of consecutive initial points are distinct, then this is true for all subdivision levels j. If we look at surfaces, this is not enough.

If we look at figure 4.5, we see a sketch of how the new adjoining points can become equal. In the figure, we have marked some of the points on level j with large dots, while some of the new points on level $j+1$ related to the points on level j are marked with small dots. The three curves are the cubic interpolants to the points needed to calculate $\mathbf{p}_{j+1,2k+1,2l}$, $\mathbf{p}_{j+1,2k+1,2l+1}$ and $\mathbf{p}_{j+1,2k+1,2l+2}$. In the sketch, these three new points are the same, denoted by $\mathbf{q}$.

If we now want to calculate a new parameterization for level $j+1$ using the distance between the points, as in a chordal or centripetal parameterization, we would get several equal parameter values. Parametric interpolation, as used in the calculation of new points, is only defined for distinct parameter values, so the scheme is not well defined in this case. Semi-regular approaches, as both the 16-point scheme and the 12-point scheme, do not have this problem, as they do not use the new points to find a new parameterization.

Chapter 5

Numerical results

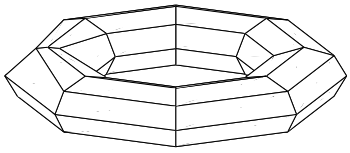
5.1 Numerical comparison

In figure 3.3, 4.1 and 4.4 we see respectively the results of the tensor product methods, the 16-point scheme and the 12-point scheme on the same initial data. In figure 5.1 we have taken cross sections of the results at the nonuniform part of the torus. Again, the 16-point and 12-point schemes gives results that are more in line with what we would expect from the initial torus.

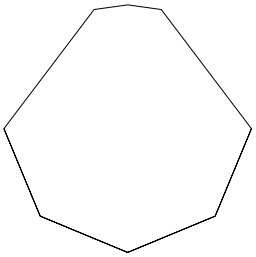
All the points on the nonuniform cross section is on a circle, so we would hope that the methods would give us results that are close to a circle. In figure 5.2 we can see how close the different schemes are to the inscribing circle of the initial cross section. The 16-point subdivision scheme and the 12-point interpolatory scheme are equal at this cross section, so only the 16-point scheme is shown.

We can numerate the interval between the points in the nonuniform cross section in figure 5.1(b) from 1 to 8, letting the bottom left be labeled 1, and going clockwise. We can now look at the maximal distance from the circle inscribing the initial cross section to the cross sections of the results of the different methods for each interval. The result of this is in table 5.1. The results for the 16-point and 12-point scheme are equal in this case, so only results for the 16-point scheme is shown. The new methods are closer to the circle than the tensor product methods.

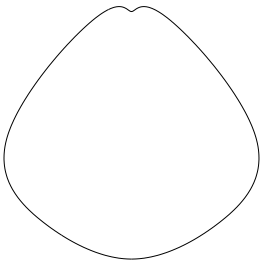
We could also do the the same for the cross section opposite of the nonuniform part. This is in table 5.2. Again, the 16-point and the 12-point scheme gives the same result, and this is also equal to the tensor product subdivision scheme. This is because all of the subdivision schemes uses the four point scheme on this initial data, the 16-point and 12-point scheme do this because they use uniform parameterization since the points are uniformly distributed. The large distance for the spline surface in 1 and 8 probably arrives from my boundary conditions, but the more interesting



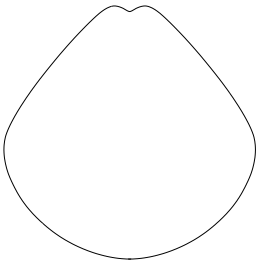
(a) Initial polygon.



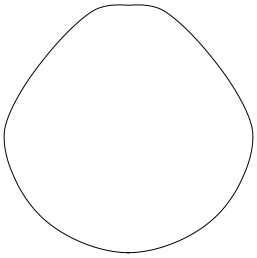
(b) A cross section of the nonuniform part of the initial polygon.



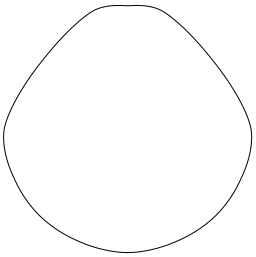
(c) A cross section of the a tensor product C^2 interpolatory spline surface.



(d) A cross section of the result of uniform tensor product interpolatory subdivision.

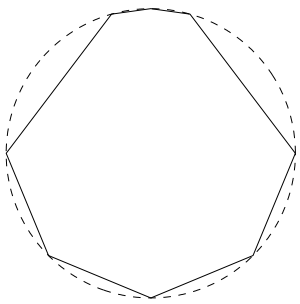


(e) A cross section of the result of the 16-point interpolatory scheme.

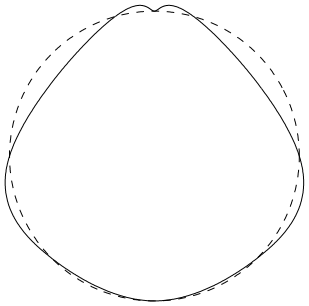


(f) A cross section of the result of the 12-point interpolatory scheme.

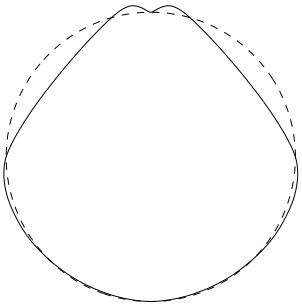
Figure 5.1: Cross sections at the nonuniform part of a numerical example with the same initial data as in figure 3.3.



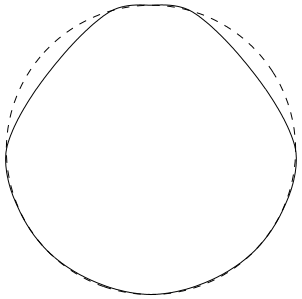
(a) Initial cross section and inscribing circle.



(b) Inscribing circle and the cross section of tensor product spline intepolation.



(c) Inscribing circle and the cross section of uniform tensor product subdivision.



(d) Inscribing circle and the cross section of the 16-point subdivision scheme.

Figure 5.2: Cross sections at the nonuniform part of a numerical example with the the inscribing circle of the initial cross section.

	Initial polygon	Spline surface	Uniform TP	16-point
1	0.07612	0.02236	0.008537	0.008474
2	0.07612	0.06263	0.0309	0.01034
3	0.2027	0.1335	0.1503	0.1198
4	0.009431	0.08351	0.06847	0.01117
5	0.009431	0.08351	0.06847	0.01117
6	0.2027	0.1335	0.1503	0.1198
7	0.07612	0.06263	0.0309	0.01034
8	0.07612	0.02236	0.008537	0.008474

Table 5.1: The maximum distance from the cross sections in figure 5.1 to the circle inscribing figure 5.1(b).

observation is that the error for interval 4 and 5 are larger than those for interval 2 and 3. This is because the spline surface does not use a uniform parameterization here, since it is the average of 7 uniform and 1 nonuniform parameterizations.

	Initial polygon	Spline surface	Uniform TP	16-point
1	0.07612	0.01217	0.008471	0.008471
2	0.07612	0.004056	0.008471	0.008471
3	0.07612	0.005674	0.008471	0.008471
4	0.07612	0.006467	0.008471	0.008471
5	0.07612	0.006467	0.008471	0.008471
6	0.07612	0.005674	0.008471	0.008471
7	0.07612	0.004056	0.008471	0.008471
8	0.07612	0.01217	0.008471	0.008471

Table 5.2: The maximum distance from the cross sections opposite of the nonuniform cross section in figure 5.1 to the circle inscribing this cross section.

Numerical examples also show small differences between the 16-point and 12-point schemes for nonuniform data, much smaller than that of the difference between these two and the tensor product methods. For uniform data the 16-point subdivision scheme equals the tensor product subdivision, but also the 12-point scheme has a smaller distance to the tensor product method in this case.

For instance, for the nonuniform torus, the maximum distance between the 16-point scheme and the uniform tensor product subdivision scheme is equal to the maximal distance between the 12-point scheme and tensor product subdivision scheme. Further, the maximum distance between the 12-point and 16-point schemes are about one third of the distance to the tensor product subdivision scheme. For a uniform torus with almost the

same data points, the maximal distance between the 12-point and 16-point scheme is reduced by about a third.

5.2 Numerical convergence

When using any subdivision scheme numerically, we only calculate points. So the curve or surface we create, is a linear interpolant g_j or a bilinear interpolant $\mathbf{g}_j$ to the points we have calculated. If we are working with curves, we know from results stated in chapter 2, that the linear interpolant g_j is an approximation to the limit g, and that the approximation gets better as j, and the number of points, increases. Similar statements can be made about the tensor product subdivision method described in section 3.2, but we have no proof that this holds for the two new methods discussed in the previous chapter.

The bilinear interpolant $\mathbf{g}_j$ is continuous by definition, so all numerical and visual inspections should state this. But this does not give any real information about the limit, so a more interesting question is how $\mathbf{g}_j$ relates to $\mathbf{g}_{j+1}$. I have not succeeded in showing that lemma 2 in chapter 2 holds for neither the 16-point nor the 12-point scheme, but we will in this section study this numerically.

We let

$$||\mathbf{f}||_\infty = \max_{x,y} |\mathbf{f}(x,y)|$$

be the max norm for functions of two variables, and $\mathbf{g}_j$ be the bilinear interpolant to the points $(\mathbf{p}_{j,k,l}, \mathbf{t}_{j,k,l})$. In chapter 2, we proved that

$$||g_{j+1} - g_j||_\infty \leq \frac{\max_k |\delta_{0,k}|}{8} \left(\frac{5}{8}\right)^j$$

holds for all $j \geq 0$ for the four point scheme. Lets now look at

$$||\mathbf{g}_{j+1} - \mathbf{g}_j||_\infty$$

for the uniform tensor product subdivision scheme. As in section 2.3.1, because of the bilinearity of $\mathbf{g}_{j+1}$ and $\mathbf{g}_j$, the maximum difference occurs at one of the new points at level $j+1$. Dyn et al. [3] proved that for the four point scheme,

$$d_{j,k} = g_{j+1,2k+1} - \frac{g_{j,k} + g_{j,k+1}}{2} \leq \frac{1}{8} \max\{||\delta_{j,k-1}||, ||\delta_{j,k+1}||\} \tag{5.1}$$

Remember, from section 2.3.1, $\delta_{j,k} := g_{j,k} - g_{j,k-1}$. Since, in the uniform tensor product scheme, we use the four point scheme in each direction, we

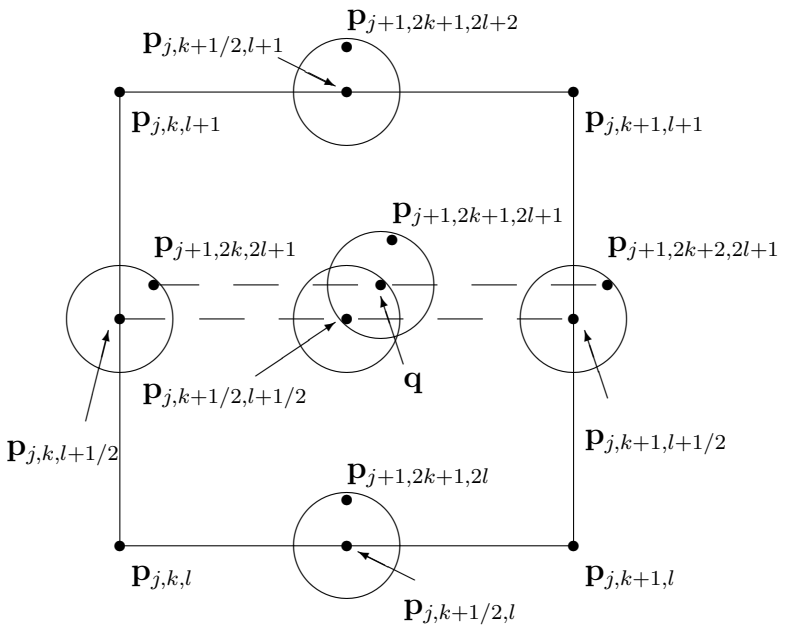


Figure 5.3: Sketch to show the distance between points at level $j+1$. The circles are all of the same size, with radius $\frac{1}{8}\max_{k,l}|\delta_{j,k,l}|$. Here $\mathbf{q} = (\mathbf{p}_{j+1,2k,2l+1} + \mathbf{p}_{j+1,2k+2,2l+1})/2$, and $\mathbf{p}_{j,k+1/2,l} = (\mathbf{p}_{j,k,l} + \mathbf{p}_{j,k+1,l})/2$.

get that the same holds for the new edgepoints. But since for the facepoint, we use the four point scheme on the new edgepoints, the difference is doubled. So, the tensor product generalization of (2.5) becomes

$$||\mathbf{g}_{j+1} - \mathbf{g}_j||_\infty \leq \frac{1}{4}\max_{k,l}|\delta_{j,k,l}|$$

where $\delta_{j,k,l}$ is the maximum of distance from $\mathbf{p}_{j,k,l}$ to $\mathbf{p}_{j,k+1,l}$ and $\mathbf{p}_{j,k,l+1}$, or

$$\delta_{j,k,l} = \max\left\{||\mathbf{p}_{j,k+1,l} - \mathbf{p}_{j,k,l}||, ||\mathbf{p}_{j,k,l+1} - \mathbf{p}_{j,k,l}||\right\}$$

In figure 5.3 we see a sketch of how the distances $\delta_{j+1,2k,2l}$ relate to $\delta_{j,k,l}$. We see that the distance between a vertex and an edgepoint is limited by $\frac{5}{8}\max_{k,l}|\delta_{j,k,l}|$, as we know from the univariate case in (2.8). For the maximum distance between an edgepoint and a facepoint, we see that this distance is bounded by $\frac{7}{8}\max_{k,l}|\delta_{j,k,l}|$. Hence,

$$||\mathbf{g}_{j+1} - \mathbf{g}_j||_\infty \leq \frac{\max_{k,l}|\delta_{0,k,l}|}{4}\left(\frac{7}{8}\right)^j$$

holds for the uniform tensor product scheme. This is enough to apply lemma 2 in chapter 2. So we can also use this to do a numerical comparison

to the convergence of 16- and 12-point subdivision schemes described in the previous chapter.

5.2.1 Numerical convergence or the 16-point scheme

In table 5.3 we se the convergence of the 16-point interpolatory subdivision scheme on the torus used in figure 4.1. Also shown is

$$\frac{\max_{k,l} |\delta_{0,k,l}|}{4} \left(\frac{7}{8}\right)^j$$

As we can see, for this dataset the convergence is much faster than what we proved to hold for the regular scheme, at least for $j \leq 10$.

j	$\|\|\mathbf{g}_{j+1} - \mathbf{g}_j\|\|_\infty$	$C_{16}\lambda_{16}^j$
0	0.331899	0.765367
1	0.09987	0.669696
2	0.0351121	0.585984
3	0.0122423	0.512736
4	0.00383783	0.448644
5	0.00114149	0.392564
6	0.00032965	0.343493
7	9.34033e-05	0.300556
8	2.6101e-05	0.262987
9	7.21496e-06	0.230114
10	1.9767e-06	0.201349

Table 5.3: The convergence of the 16-point scheme on the nonuniform torus, where $C_{16} = \frac{\max_{k,l} |\delta_{0,k,l}|}{4}$ and $\lambda_{16} = \frac{7}{8}$.

With one iteration of a surface subdivision scheme on $n \times m$ points, the result is $(2n-1) \times (2m-1)$ points. So we after each iteration of a scheme, we have almost four times the number of points as before the iteration. Hence, after ten steps we have almost one million times more points than the initial data, so doing more iterations for all of the points needs more storage and computing capacity than we have available.

Visually and numerically one seldom needs to do ten iterations to get good results, as $||\mathbf{g}_{j+1} - \mathbf{g}_j||_\infty$ is negligible when j gets close 10. Also, the distance between points, $|\delta_{j,k,l}|$, is getting smaller fast as j increases, for the example in table 5.3, $\max_{k,l} |\delta_{10,k,l}|$ is about thousand times smaller than $\max_{k,l} |\delta_{0,k,l}|$.

By studying numerical results, it seems as the points where we have the maximum $||\mathbf{g}_{j+1} - \mathbf{g}_j||_\infty$ occurs near the point where we have the maximum $||\mathbf{g}_j - \mathbf{g}_{j-1}||_\infty$. So we can use this to overcome the the storage and computing

j	$\|\|\mathbf{g}_{j+1} - \mathbf{g}_j\|\|_\infty$	$C_{16}\lambda_{16}^j$
0	0.331899	0.765367
5	0.00114149	0.392564
10	1.9767e-06	0.201349
15	2.77905e-09	0.103274
20	3.54424e-12	0.0529701
25	5.51228e-15	0.0271688

Table 5.4: The convergence of the 16-point scheme on the nonuniform torus, where we have restricted the calculations of new points to the neighbourhood of the maximum difference. Also shown is the convergence criteria fulfilled by the tensor product subdivision scheme, where $C_{16} = \frac{\max_{k,l} |\delta_{0,k,l}|}{4}$ and $\lambda_{16} = \frac{7}{8}$.

problems, by only using points in the the neighbourhood of the maximum difference between two consecutive bilinear interpolants. If we do this were we take 25 points in each direction from where the maximum difference is, we get the result in table 5.4. For the first ten iterations, the maximum error is equal, so this approximation seems like a good idea. We see that the rate of convergence is higher than what we know is enough to prove that the uniform scheme is continuous. In fact, in table 5.5 we see that the convergence of the 16-point interpolatory scheme is quite similar to the tensor product subdivision scheme. The values for the uniform tensor product scheme are calculated similarly to those for the 16-point scheme, the ten first are complete calculations, but for $j > 10$ the assumption that the biggest error occurs near the previous maximal error is used.

j	16-point scheme: $\|\|\mathbf{g}_{j+1} - \mathbf{g}_j\|\|_\infty$	Tensor product: $\|\|\mathbf{g}_{j+1} - \mathbf{g}_j\|\|_\infty$
0	0.331899	0.338663
5	0.00114149	0.00115275
10	1.9767e-06	2.08121e-06
15	2.77905e-09	2.96595e-09
20	3.54424e-12	3.80842e-12
25	5.51228e-15	4.96507e-15

Table 5.5: The convergence of the 16-point interpolatory subdivision scheme, and the convergence of the tensor product subdivision scheme on a nonuniform torus.

Another interesting example is this set of initial data in $\mathbb{R}^3$:

$$\begin{aligned} \mathbf{p}_{0,0,0} &= (0,0,1) \\ \mathbf{p}_{0,k,l} &= (k,l,0) \qquad \forall k,l \end{aligned} \tag{5.2}$$

Here all the points are on the x, y-plane, except from one, which is above it. The convergence of the 16-point method, the tensor product subdivision and the convergence property we know the tenor product method fulfills are shown in table 5.6. Again, the convergence rate for both of the schemes are quite similar, but they are both closer to the worst convergence rate the tensor product scheme can have, at least for the first few steps. It is interesting to note that $||\mathbf{g}_2 - \mathbf{g}_1||_\infty$ is larger than $||\mathbf{g}_1 - \mathbf{g}_0||_\infty$. The calculations for $j > 10$ are again based on the assumption that the largest difference occurs near the previous largest difference.

j	16-point: $\|\|\mathbf{g}_{j+1} - \mathbf{g}_j\|\|_\infty$	Tensor prod.: $\|\|\mathbf{g}_{j+1} - \mathbf{g}_j\|\|_\infty$	$C_{16}\lambda_{16}^j$
0	0.0536108	0.0664062	0.353553
1	0.100582	0.101562	0.309359
2	0.0527421	0.0511627	0.270689
3	0.020076	0.0191307	0.236853
4	0.00668028	0.00630814	0.207247
5	0.0020754	0.00194955	0.181341
10	3.97365e-06	3.69548e-06	0.0930113
15	5.77769e-09	5.3551e-09	0.0477063
20	7.495e-12	6.9349e-12	0.024469
25	1.35979e-14	8.43769e-15	0.0125504

Table 5.6: The convergence of the 16-point interpolatory subdivision scheme, the convergence of the tensor product subdivision scheme and $C_{16}\lambda_{16}^j$, where $C_{16} = \frac{\max_{k,l}|\delta_{0,k,l}|}{8}$ and $\lambda_{16} = \frac{5}{8}$ on the example in (5.2).

We have checked convergence on a number of other examples, and the difference between two bilinear interpolants $||\mathbf{g}_{j+1} - \mathbf{g}_j||_\infty$ are always less than $\frac{\max_{k,l}|\delta_{0,k,l}|}{4}\left(\frac{7}{8}\right)^j$ for the 16-point scheme.

5.2.2 Numerical convergence of the 12-point scheme

In table 5.7 we se the convergence of the 12- and 16-point interpolatory schemes for the torus in figure 4.1, so for the 16-point scheme this is the same data as the one in table 5.3. We observe that the convergence is very similar for the two schemes.

The 12-point method has the same storage and computational requirements as the 16-point method, but again we can try to assume that the maximum error is near the maximum error from the last step. This gives us the results in table 5.8. Again the results are equal for $j \leq 10$, and are quite close to the 16-point scheme, and thus comparable both to the tensor product interpolatory subdivision scheme, and the convergence criteria it fulfills.

j	12-point: $\|\|\mathbf{g}_{j+1} - \mathbf{g}_j\|\|_\infty$	16-point: $\|\|\mathbf{g}_{j+1} - \mathbf{g}_j\|\|_\infty$
0	0.326726	0.331899
1	0.0986776	0.09987
2	0.0342718	0.0351121
3	0.0121317	0.0122423
4	0.00383713	0.00383783
5	0.00114361	0.00114149
6	0.000330049	0.00032965
7	9.34372e-05	9.34033e-05
8	2.61002e-05	2.6101e-05
9	7.2143e-06	7.21496e-06
10	1.97662e-06	1.9767e-06

Table 5.7: The convergence of the 12-point interpolatory scheme and the 16-point interpolatory scheme on a nonuniform torus.

j	12-point: $\|\|\mathbf{g}_{j+1} - \mathbf{g}_j\|\|_\infty$	16-point: $\|\|\mathbf{g}_{j+1} - \mathbf{g}_j\|\|_\infty$
0	0.326726	0.331899
5	0.00114361	0.00114149
10	1.97662e-06	1.9767e-06
15	2.7791e-09	2.77905e-09
20	3.54402e-12	3.54424e-12
25	5.95833e-15	5.51228e-15

Table 5.8: The convergence of the 12-point interpolatory scheme and the 16-point interpolatory scheme on a nonuniform torus, where we have only calculated new points near the maximal error.

Again, studying the almost planar example in (5.2) with the 12-point scheme is interesting. The result of this is in table 5.9, together with the results from the 16-point scheme, and the convergence results are again quite close.

As for the 16-point scheme, we have run calculations on a number of other examples, and the convergence rate is always less than $\frac{\max_{k,l}|\delta_{0,k,l}|}{4}\left(\frac{7}{8}\right)^j$.

j	12-point: $\|\|\mathbf{g}_{j+1}-\mathbf{g}_j\|\|_\infty$	16-point: $\|\|\mathbf{g}_{j+1}-\mathbf{g}_j\|\|_\infty$
0	0.0482419	0.0536108
1	0.0966732	0.100582
2	0.0521216	0.0527421
3	0.020126	0.020076
4	0.00672083	0.00668028
5	0.00208502	0.0020754
6	0.000620493	0.000619024
7	0.000179838	0.000179685
8	5.11514e-05	5.1143e-05
9	1.43397e-05	1.43403e-05
10	3.97342e-06	3.97365e-06
15	5.77769e-09	5.77769e-09
20	7.49512e-12	7.495e-12
25	1.37614e-14	1.35979e-14

Table 5.9: The convergence of the 12-point interpolatory scheme and the 16-point interpolatory scheme on the example in (5.2), where, for $j > 10$, we have only calculated new points near the maximal error.

5.3 Numerical tests for smoothness

For a parametric surface to be C^1, we must be able to differentiate it with respect to its parameters. But since we don't have a limit surface, we have noting to differentiate. Furthermore, we do not know if the limit surface, if it exists, is differentiable with respect to the parameterization we have chosen. In fact, the limit surface might not be C^1 with respect to the parameterization we have used to calculate it, but might be C^1 with respect to another, unknown parameterization.

To overcome this we will instead study the tangent planes of the bilinear interpolant $\mathbf{g}_j$. If the tangent planes of the sequence $\mathbf{g}_0, \mathbf{g}_1, \ldots$ converge, this implies that the limit surface $\mathbf{g}$ is tangent plane continuous. This, in turn, implies that there might exists a parameterization such that the limit surface is C^1.

We are only interested in the tangent planes at the points, and the tangent plane can be expressed by one vector, the normal vector of the tangent plane. A normal vector for a point $\mathbf{p}_{j,k,l}$ can be found by taking the cross product between $\mathbf{p}_{j,k+1,l} - \mathbf{p}_{j,k,l}$ and $\mathbf{p}_{j,k,l+1} - \mathbf{p}_{j,k,l}$. This is the normal vector of the face spanned by $\mathbf{p}_{j,k,l}$, $\mathbf{p}_{j,k+1,l}$ and $\mathbf{p}_{j,k,l+1}$. But this normal vector is not symmetric, so a better approach is to take the average of the normal vectors of all the faces adjoining $\mathbf{p}_{j,k,l}$. This is

$$\begin{aligned}\mathbf{N}_{j,k,l} = \frac{1}{4}\Big(& (\mathbf{p}_{j,k+1,l} - \mathbf{p}_{j,k,l}) \times (\mathbf{p}_{j,k,l+1} - \mathbf{p}_{j,k,l}) \\ & + (\mathbf{p}_{j,k,l+1} - \mathbf{p}_{j,k,l}) \times (\mathbf{p}_{j,k-1,l} - \mathbf{p}_{j,k,l}) \\ & + (\mathbf{p}_{j,k-1,l} - \mathbf{p}_{j,k,l}) \times (\mathbf{p}_{j,k,l-1} - \mathbf{p}_{j,k,l}) \\ & + (\mathbf{p}_{j,k,l-1} - \mathbf{p}_{j,k,l}) \times (\mathbf{p}_{j,k+1,l} - \mathbf{p}_{j,k,l})\Big)\end{aligned}$$

where $\mathbf{N}_{j,k,l}$ is the normal vector at $\mathbf{p}_{j,k,l}$ and $\times$ is the cross product.

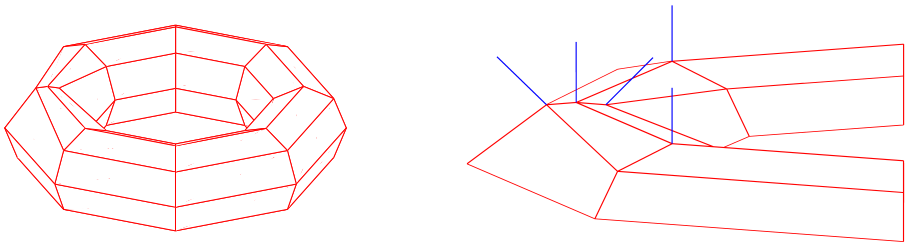
We will only study the convergence of the normal vectors at a initial point, since we in section 4.1.1 showed that in the interior of an initial face, we get a tensor product patch, which is C^1. So for a initial point $\mathbf{p}_{0,\hat{k},\hat{l}}$, we can look at the maximum angle $\theta_{\max}$ between $\mathbf{N}_{j,k,l}$ and its four neighbours $\mathbf{N}_{j,k+1,l}$, $\mathbf{N}_{j,k,l+1}$, $\mathbf{N}_{j,k-1,l}$ and $\mathbf{N}_{j,k,l-1}$, where $\mathbf{p}_{j,k,l} = \mathbf{p}_{0,\hat{k},\hat{l}}$.

5.3.1 Tangent plane continuity for the 16-point subdivision scheme

In figure 5.4 we see the torus we have used in several examples, and in figure 5.4(b) we see a part of this torus, with five normal vectors. We let the one in the middle be $\mathbf{N}_{0,k,l}$. The maximum angle $\theta_{\max}$ for the 16-point interpolatory subdivision scheme and for the uniform tensor product scheme for this example is shown in table 5.10. We observe that the tangent plane of the 16-point scheme seems to converge faster than that of the uniform tensor product subdivision scheme for this example.

j	16-point $\theta_{\max}$	Tensor prod. $\theta_{\max}$
0	0.525292	0.525292
5	0.0250339	0.251836
10	0.00210614	0.0187476
15	0.000107178	0.000918034
20	4.64202e-06	3.90653e-05
25	1.54139e-07	1.52633e-06

Table 5.10: The maximum angle between a normal vector and its four neighbours for the 16-point subdivision scheme and for the uniform tensor product scheme.



(a) Initial polygon.

(b) A part of the initial polygon with normal vectors.

Figure 5.4: A numerical example with the same initial data as in Figure 3.3, and normal vectors at five points.

A comparison with tensor product interpolating splines are not quite as simple, since we for a spline surface calculate a formula for the surface, rather than explicit points as with subdivision. An acceptable approximation is to use parameter values. For the subdivision schemes we know that $\mathbf{p}_{j,k,l}$ and $\mathbf{p}_{j,k+1,l}$ has parameter values $\mathbf{t}_{j,k,l} = (u_{j,k,l}, v_{j,k,l})$ and $\mathbf{t}_{j,k+1,l} = (u_{j,k+1,l}, v_{j,k+1,l})$. If $\mathbf{p}_{j,k,l} = \mathbf{p}_{0,\hat{k},\hat{l}}$, then

$$u_{j,k+1,l} - u_{j,k,l} = \frac{u_{0,\hat{k}+1,\hat{l}} - u_{0,\hat{k},\hat{l}}}{2^j}$$

We can use this to find points on a parametric spline surface to use in comparisons with subdivision methods. So we are given a parametric spline surface $\mathbf{g}(r,s)$ which interpolate $\mathbf{p}_{0,\hat{k},\hat{l}}$ and $\mathbf{p}_{0,\hat{k}+1,\hat{l}}$ with parameter values $(r_{\hat{k}}, s_{\hat{l}})$ and $(r_{\hat{k}+1}, s_{\hat{l}})$. We have a subdivision method who at the same data has calculated $\mathbf{p}_{j,k+1,l}$, where $\mathbf{p}_{j,k,l} = \mathbf{p}_{0,\hat{k},\hat{l}}$, and want to find an approximation to this point on the spline surface. This approximation is then given by

$$\mathbf{g}\Big(\frac{t_{\hat{k}+1} - t_{\hat{k}}}{2^j}, s_{\hat{l}}\Big)$$

We have used this to find the values in table 5.11, and we can see that the 16-point interpolatory scheme compares favourably with the tensor product interpolating spline surface. What this really means, is that if we calculate a subdivision surface with a number of points, this surface will, for this example, appear at least as smooth as a spline surface with the same number of points.

We could also do exactly the same for an uniform torus. The result of this is in table 5.12. Not surprisingly, the 16-point scheme and the uniform

j	16-point $\theta_{\max}$	Spline $\theta_{\max}$
0	0.525292	0.525292
5	0.0250339	0.175977
10	0.00210614	0.00604967
15	0.000107178	0.000189457
20	4.64202e-06	5.9215e-06
25	1.54139e-07	2.07549e-07

Table 5.11: The maximum angle between a normal vector and its four neighbours for the 16-point subdivision scheme and for a tensor product interpolating spline surface for the nonuniform torus in figure 5.4.

tensor product subdivision scheme are equal for this example, as the 16-point scheme reduces to the tensor product in this uniform case. The tensor product splines are however a little bitt better, but the difference from the nonuniform torus is smallest for the 16-point scheme.

j	16-point $\theta_{\max}$	Tensor prod. $\theta_{\max}$	Spline $\theta_{\max}$
0	0.773819	0.773819	0.773819
5	0.037391	0.037391	0.0257082
10	0.0017085	0.0017085	0.000808043
15	7.02561e-05	7.02561e-05	2.5256e-05
20	2.72276e-06	2.72267e-06	7.8948e-07
25	1.03238e-07	1.03238e-07	2.58096e-08

Table 5.12: The maximum angle between a normal vector and its four neighbours for the 16-point subdivision scheme, for an uniform tensor product subdivision scheme and for an interpolating spline surface on an uniform torus.

We have also done similar tests for the example in (5.2), where we check the convergence of the tangent planes at $\mathbf{p}_{0,0,0}$, the one point above the x, y-plane. The results for this is in table 5.13. Here, the 16-point subdivision scheme has the slowest convergence of the tangent planes, but it is very similar to the convergence of the uniform tensor product interpolatory subdivision scheme.

5.3.2 Tangent plane continuity for the 12-point interpolatory surface scheme

In this section we will do the same numerical tests for the 12-point interpolatory scheme as we did for the 16-point scheme in the previous section. We start by the well known torus in figure 5.4. The maximal angle between

j	16-point $\theta_{\max}$	Tensor prod. $\theta_{\max}$	Spline $\theta_{\max}$
0	0.392699	0.392699	0.392699
5	0.26656	0.242759	0.131778
10	0.0168845	0.0151349	0.00430157
15	0.000785719	0.000701904	0.000134564
20	3.26163e-05	2.90871e-05	4.20543e-06
25	1.27359e-06	1.13249e-06	1.35756e-07

Table 5.13: The maximum angle between a normal vector and its four neighbours for the 16-point subdivision scheme, for an uniform tensor product subdivision scheme and for an interpolating spline surface for the example in (5.2).

the normal vector at the point between the nonuniform data and its neighbours are shown in table 5.14. From the table it seems as the two schemes have exactly the same convergence of the tangent planes, and this is the case for a significant number of digits. For $j = 1$, the maximum angle $\theta_{\max}$ is 0.1329537 for the 12-point scheme, and the difference in maximum angle between the 12-point and the 16-point scheme is -9.190397e-07, so the 12-point scheme has a slightly better convergence of the normal vectors.

j	12-point $\theta_{\max}$	16-point $\theta_{\max}$
0	0.525292	0.525292
5	0.0250339	0.0250339
10	0.00210614	0.00210614
15	0.000107178	0.000107178
20	4.64202e-06	4.64202e-06
25	1.54139e-07	1.54139e-07

Table 5.14: The maximum angle between a normal vector and its four neighbours for the 12-point scheme and for the 16-point scheme for the example in figure 5.4.

In table 5.15 we do the same numerical tests on an uniform torus, so the results are comparable with table 5.12. Again, the convergences of the tangent planes are almost similar for the 12-point scheme and the 16-point scheme.

We have also done numerical tests for the tangent plane convergence of 12-point scheme on the example in (5.2), and the results are in table 5.16. Here, the 12-point scheme has a convergence of the tangent plane that is slightly worse than that of the 16-point scheme for small j, but for larger j the maximal angle between the normal vectors is equal.

j	12-point $\theta_{\max}$	16-point $\theta_{\max}$
0	0.773819	0.773819
5	0.037391	0.037391
10	0.0017085	0.0017085
15	7.02561e-05	7.02561e-05
20	2.72276e-06	2.72276e-06
25	1.03238e-07	1.03238e-07

Table 5.15: The maximum angle between a normal vector and its four neighbours for the 12-point scheme and for the 16-point scheme for an uniform torus.

j	12-point $\theta_{\max}$	16-point $\theta_{\max}$
0	0.392699	0.392699
5	0.266562	0.26656
10	0.0168845	0.0168845
15	0.000785719	0.000785719
20	3.26163e-05	3.26163e-05
25	1.27359e-06	1.27359e-06

Table 5.16: The maximum angle between a normal vector and its four neighbours for the 12-point scheme and for the 16-point scheme for the example in (5.2).

Chapter 6

Conclusions

The two new methods, the 16-point interpolatory subdivision scheme and the 12-point interpolatory scheme, both give more visually pleasing results for nonuniformly spaced data than the well known interpolatory tensor product methods for splines and subdivision. Numerical examples also show that the limit surfaces we get from the 16-point and 12-point schemes on nonuniform data are closer to the surfaces we get from similar, uniform data than the tensor product methods.

The 16-point and 12-point schemes are both continuous and differentiable for uniform data, we know this because they then equal other, previously know methods. Numerical tests and examples suggest that they also are continuous and differentiable for nonuniform data.

If we need surfaces that are two times differentiable, the new methods can be extended to use an interpolatory curve subdivision method that uses six points instead of four. Such curve methods exist, and some are shown to be C^2, and hopefully this will give surface methods that are C^2.

Further work could include generalizing a proof for continuity and differentiability of semiregular/irregular interpolatory curve subdivision to surfaces, or to find a new proof for the smoothness of the subdivision surfaces. One could try to generalize the methods discussed here to use irregular subdivision, to use different parameterizations, or to use more points for a (hopefully) better limit surface.

Bibliography

[1] Daubechies, I., Guskov, I., Sweldens, W.: "Regularity of Irregular Subdivision", *Conts. Approx.* 15, (1999): 381-426

[2] Dubuc, S.: "Interpolation through an Iterative Scheme", *Journal of Mathematical Analysis and Applications* 114, (1986): 185-204

[3] Dyn, N., Floater, M.S., Hormann, K.: "Four-point curve subdivision based on iterated chordal and centripetal parameterizations", *Computer Aided Geometric Design* 26, (2009): 279-286

[4] Dyn, N., Levin, D., Gregory, J.: "A 4-point Interploatory Subdivision Scheme for Curve Design", *Computer Aided Geometric Design* 4, (1987): 257-268

[5] Farin, G.: *Curves and surfaces for CAGD*, fifth edition, San Francisco: Morgan Kaufmann, 2002

[6] Floater, M.: "A piecewise polynomial approach to analyzing interpolatory subdivision", preprint,http://folk.uio.no/michaelf/papers/ppa.pdf

[7] Floater, M.: "The approximation order of four-point interpolatory curve subdivision", to appear in J. Comp. Appl. Math, http://folk.uio.no/michaelf/papers/sub_param.pdf

[8] Floater, M.: Lecture notes in INF4360 - Topics in Geometric modelling, http://www.uio.no/studier/emner/matnat/ifi/INF4360/

[9] Hed, S.: *Analysis of Subdivision Schemes for Surface Generation.* Master thesis, Tel-Aviv University, 1992

[10] Lyche, T., Mørken, K.: "Spline Methods Draft", lecture notes in INF-MAT5340, http://www.uio.no/studier/emner/matnat/ifi/INF-MAT5340/v11/undervisningsmateriale/book.pdf

[11] Warren, J.: "Binary Subdivision Schemes for Functions over Irregular Knot Sequences", in *Mathematical Methods in CAGD III*, Daehlen, M., Lyche, T., Schumaker, L. (eds.), Academic Press, New York (1995): 543-562